DFSMSrmm Primer

- Learn how DFSMSrmm manages your tape environment
- Discover the latest DFSMSrmm enhancements
- Find DFSMSrmm implementation details

Mary Lovelace
Norbert Schlumberger

ibm.com/redbooks

Redbooks

International Technical Support Organization

DFSMSrmm Primer

March 2014

SG24-5983-04

Note: Before using this information and the product it supports, read the information in “Notices” on page xiii.

Fourth Edition (March 2014)

This edition applies to Version 1, Release 13 of z/OS DFSMSrmm (product number 5694-A01).

Contents

Notices . . . xiii
Trademarks . . . xiv

Preface . . . xv
Authors . . . xv
Now you can become a published author, too! . . . xvi
Comments welcome . . . xvi
Stay connected to IBM Redbooks . . . xvii

Chapter 1. DFSMSrmm enhancements . . . 1
1.1 Changes introduced with z/OS V1R13 DFSMSrmm . . . 2
1.1.1 Customize point-and-shoot fields in the DFSMSrmm dialog . . . 2
1.1.2 Specifying the retention method to use for new tape data sets . . . 2
1.1.3 IBM TS1140 support . . . 2
1.1.4 Copying data set attributes in a tape copy application . . . 2
1.1.5 Excluding specific data sets from VRSEL processing . . . 3
1.1.6 Inventory management considerations for EXPROC and VRSEL . . . 3
1.1.7 Volume usage, capacity, and compression considerations . . . 3
1.2 Changes introduced with z/OS V1R12 DFSMSrmm . . . 4
1.2.1 Ease of use and flexibility enhanced . . . 4
1.2.2 Simplification . . . 5
1.2.3 Storage reporting . . . 5
1.2.4 Availability . . . 5
1.2.5 Scalability and performance . . . 5
1.3 Changes introduced with z/OS V1R11 DFSMSrmm . . . 5
1.3.1 EDGINERS tape label scan . . . 5
1.3.2 Dynamic installation exits . . . 6
1.3.3 Returning volumes to the system-managed library . . . 6
1.3.4 VRSEL GDG option . . . 7
1.3.5 DFSMSrmm Report Generator . . . 7
1.3.6 DFSMSrmm usability items . . . 8
1.3.7 VRSEL(OLD) parmlib option . . . 8
1.4 Changes introduced with z/OS V1R10 DFSMSrmm . . . 8
1.4.1 Enable use of DFSMSdss COPY Service . . . 9
1.4.2 SMF record changes . . . 9
1.4.3 Audit controls for release processing . . . 9
1.4.4 XREPTEXT tailoring via SYSIN . . . 10
1.4.5 Volume replacement policies . . . 10
1.4.6 Report Generator enhancements . . . 10
1.4.7 System-managed library partitioning . . . 11
1.4.8 Common Information Model (CIM) Provider . . . 12
1.5 Changes introduced with z/OS V1R9 DFSMSrmm . . . 12
1.5.1 Task management support . . . 12
1.5.2 Multitasking of utilities . . . 12
1.5.3 Control data set (CDS) serialization . . . 12
1.5.4 JCL data set names . . . 13
1.5.5 Shared parmlib support . . . 13
1.5.6 TSO subcommands . . . 13
1.5.7 3592 Model E05 software support . . . 13

1.5.8 REPORT17 of EDGRRPTE REXX Exec 14
1.5.9 Common Information Model (CIM) provider 14
1.6 Changes introduced with z/OS V1R8 DFSMSrmm 15
1.6.1 Support for a true email address for the RMM NOTIFY function 15
1.6.2 Setting up DFSMSrmm common time support 15
1.6.3 DFSMSrmm VRS policy management simplification 15
1.6.4 DFSMSrmm usability items 16
1.6.5 Tape data set authorization 16
1.7 Changes introduced with z/OS V1R7 DFSMSrmm 16
1.7.1 Issue DFSMSrmm TSO commands from the console 16
1.7.2 Improved security control over DFSMSrmm functions 16
1.7.3 Use of large format data sets 17
1.7.4 Enterprise enablement 17
1.8 Changes introduced with z/OS V1R6 DFSMSrmm 17
1.8.1 ISPF dialog enhancements 17
1.8.2 DFSMSrmm client/server support 18
1.8.3 Enhanced IBM 3592 support 18
1.8.4 Command reference summary moved to DFSMSrmm guide 18
1.8.5 C++ API with XML output option 18
1.9 Changes introduced with z/OS V1R5 DFSMSrmm 18
1.9.1 Enhanced multilevel security 18
1.10 Changes introduced with z/OS V1R4 DFSMSrmm 19
1.10.1 Back up the DFSMSrmm control data set and journal at any time 19
1.10.2 Checking authorization to ignore volumes 19
1.10.3 Duplicate volume support 19
1.10.4 Enhanced multi-level security 19
1.11 Changes introduced with z/OS V1R3 DFSMSrmm 20
1.11.1 CDS utilization displays now available 20
1.11.2 Journal utilization and status displays now available 20
1.11.3 Movement date maintained for ON LOAN volumes 20
1.11.4 Enhanced EJECT support 20
1.11.5 Enhanced foreign tape authorization 20
1.11.6 3590-H support 20
1.11.7 Report Generator enhancements 20
1.11.8 Expiration Date field added to DFSMSrmm Data Set record 21
1.11.9 Multivolume dialog support 21
1.11.10 Special character support 21
1.11.11 Changed messages for improved diagnostics 22
1.11.12 HELP moved from SYS1.SEDGHLP1 to SYS1.HELP 22

Chapter 2. Introduction to DFSMSrmm 23
2.1 Environment 24
2.1.1 CDS backup and recovery 25
2.2 Reports 25
2.2.1 Security 27
2.3 How is DFSMSrmm structured 27
2.4 What are the DFSMSrmm interfaces 29
2.4.1 Software interfaces 29
2.4.2 User interfaces 33
2.4.3 Application programming interface 35
2.4.4 TCP/IP interface 35
2.5 Library and storage location management 36
2.5.1 Removable media library 37

2.5.2 Storage location 42
2.6 HOME location 42
2.7 Media types 43
2.7.1 Tape Cartridges used in the 3592 Tape Drives 45
2.7.2 IBM 3592 Tape Cartridge read and write formats 47
2.7.3 DFSMSrmm IBM media information 47
2.7.4 IBM 3592 cartridge memory 48
2.8 Volume management 48
2.8.1 Types of volumes 48
2.8.2 Volume status 49
2.8.3 Validation 51
2.8.4 Ignore volume support 52
2.8.5 Support for duplicate volumes 54
2.9 Policies for retention and movement 56
2.9.1 Data set retention 56
2.9.2 Volume retention 58
2.9.3 Movement 59
2.10 SMS ACS support 60
2.11 DFSMSrmm automated tape library support 60
2.11.1 Cartridge entry processing 62
2.11.2 Ejecting volumes from system-managed libraries 63
2.11.3 Volume-not-in-library processing 63
2.11.4 Verifying the databases 63
2.11.5 VTS import/export processing 64
2.11.6 Defining stacked volumes to DFSMSrmm 64
2.12 Catalog synchronization 65
2.13 Implementing the RMMplex 66
2.13.1 RMMplex 66

Chapter 3. Preparing the environment 69
3.1 DFSMSrmm implementation 70
3.1.1 z/OS DFSMSrmm Customization Wizard 70
3.1.2 Updating the installation-wide exits 71
3.1.3 Implementing JES3 USERMODs 72
3.1.4 Updating and validating the PARMLIB members 72
3.1.5 Enabling ISPF Data Set List (DSLIST) support 116
3.1.6 REXX Variable Constraint Relief 117
3.1.7 Creating a starting procedure 118
3.1.8 Defining an alias for high-level qualifier RMM 121
3.1.9 Protecting DFSMSrmm resources 121
3.1.10 Defining RACF resources and groups for DFSMSrmm 122
3.1.11 Creating the DFSMSrmm CDS 123
3.1.12 Creating the DFSMSrmm CDS as extended format 126
3.1.13 Creating the DFSMSrmm journal 127
3.1.14 Restarting MVS with DFSMSrmm implemented 129
3.1.15 Starting and restarting DFSMSrmm 130
3.1.16 Stopping DFSMSrmm 132
3.1.17 Setting up DFSMSrmm utilities 134
3.1.18 Running the installation verification program 134
3.2 Tailoring the DFSMSrmm ISPF dialog 134
3.2.1 Adding DFSMSrmm to ISPF panels 135
3.2.2 Modifying the DFSMSrmm panel 136
3.2.3 Other changes 137

3.3 Making TSO HELP information available to users 138
3.4 Verifying your DFSMSrmm implementation 138
3.4.1 Displaying the parmlib options and control information. 138
3.4.2 Adding owner information 141
3.4.3 Adding racks to DFSMSrmm. 143
3.4.4 Adding volumes to DFSMSrmm 144
3.4.5 Adding bins to DFSMSrmm. 145
3.4.6 Adding a DSNAME VRS to DFSMSrmm 146
3.4.7 Creating tape data sets. 146
3.4.8 Allocating inventory management data sets 147
3.4.9 Running inventory management 148
3.4.10 Confirming movements 149
3.4.11 Creating reports 150
3.4.12 Running EDGINERS to initialize volumes automatically. 151
3.4.13 Restoring the CDS 152
3.4.14 Verifying the CDS 155
3.4.15 Testing implemented exits 155
3.4.16 Testing application use of tape 158
3.5 Education 159
3.6 New operator procedures 159

Chapter 4. DFSMSrmm retention methods 161
4.1 Overview 162
4.2 Specifying a retention method. 162
4.2.1 Expiration date 162
4.2.2 Retention date. 162
4.2.3 Using the EDGRMMnn parmlib option RETENTIONMETHOD. 163
4.2.4 EDG_EXIT100 retention method support 164
4.2.5 Subcommands for RETENTIONMETHOD parameters 168
4.2.6 Syntax format of the RETENTIONMETHOD operand 168
4.2.7 Using the SEARCHVOLUME subcommand 169

Chapter 5. Excluding data sets from VRSEL processing 175
5.1 Overview 176
5.2 Subcommands for VRSELEXCLUDE parameters 176
5.2.1 Syntax format of the VRSELEXCLUDE operand 176
5.2.2 Using EDG_EXIT100 to exclude data sets from VRSEL support 177
5.2.3 Using the SEARCHDATASET subcommand 180
5.2.4 Inventory Management VRSEL/EXPROC Processing 185

Chapter 6. Setting up an RMMplex 191
6.1 RMMplex 192
6.1.1 Maintenance levels 193
6.2 Client/server terminology 193
6.2.1 DFSMSrmm server 193
6.2.2 SERVERNAME. 193
6.2.3 PORT 194
6.2.4 SERVERTASKS operand 194
6.2.5 DFSMSrmm client. 194
6.2.6 LOCALTASKS 194
6.3 Implementing client and server systems 194
6.3.1 Checking your TCP/IP configuration 195
6.3.2 EDGRMMxx DFSMSrmm options. 200
6.3.3 Updating EDGRMMxx for a client 202

6.3.4 Updating EDGRMMxx for a server . . . 202
6.4 Updating the DFSMSrmm CDSID (optional) . . . 203
6.4.1 Sample EDGUTIL job control . . . 203
6.5 Updating your firewall (optional) . . . 204
6.6 Starting DFSMSrmm in an RMMplex . . . 204
6.6.1 Starting and restarting DFSMSrmm on your client and server . . . 204
6.7 Managing volumes in an RMMplex . . . 206
6.7.1 Catalogs in an RMMplex . . . 206
6.7.2 RACF considerations in an RMMplex . . . 207
6.7.3 System-managed libraries in an RMMplex . . . 208
6.7.4 HOUSEKEEP processing in an RMMplex . . . 208
6.8 Operator commands . . . 209

Chapter 7. Tailoring your DFSMSrmm environment . . . 211
7.1 IBM Tivoli Workload Scheduler . . . 212
7.1.1 IBM Tivoli Workload Scheduler overview . . . 212
7.1.2 Scheduling your inventory management tasks . . . 213
7.1.3 Preparation . . . 215
7.1.4 CDS verify . . . 215
7.1.5 CDS backup . . . 215
7.1.6 Inventory management . . . 215
7.1.7 Erasing and labeling volumes . . . 217
7.1.8 Scratch processing . . . 217
7.1.9 Scratch reporting . . . 217
7.1.10 Ejecting volumes from a system-managed tape library . . . 218
7.1.11 Producing reports . . . 218
7.1.12 Move confirmation . . . 219
7.1.13 Additional considerations . . . 219
7.2 Setting up DFSMSrmm common time support . . . 220
7.2.1 How DFSMSrmm uses the date and time . . . 221
7.2.2 Date and time in a DFSMSrmm client/server environment . . . 221
7.2.3 Enable common time support . . . 222
7.2.4 Potential problems using local time . . . 225
7.3 Dynamic installation exits . . . 225
7.3.1 DFSMSrmm installation exit calls . . . 226
7.3.2 Managing exit modules . . . 227
7.3.3 Setting Up the EDG_EXIT100 routine environment . . . 230
7.3.4 Controlling the exit routine through the z/OS dynamic exits facility . . . 232
7.3.5 Deleting the EDG_EXIT100 exit routines . . . 233
7.3.6 Writing an exit routine for the EDG_EXIT100 exit . . . 233
7.4 Maintaining your control data set . . . 234
7.4.1 Using EDGHSKP . . . 234
7.4.2 Using EDGBKUP . . . 237
7.4.3 EDGUTIL . . . 243
7.5 Initializing and erasing volumes . . . 245
7.6 Disposition processing . . . 250
7.7 Software products and foreign tapes . . . 251
7.7.1 Defining software products . . . 251
7.7.2 Foreign tapes . . . 252
7.7.3 Bypass DFSMSrmm and storage management subsystem processing . . . 254
7.8 DFSMSrmm volume pools . . . 254
7.8.1 Defining pools . . . 255
7.8.2 Scratch pooling . . . 257

7.9 Deleting non-existent volumes 257
7.10 Repairing actions 258
7.10.1 Using the VERIFY parameter 259
7.10.2 Using the MEND parameter 260
7.10.3 Using EDGSPLCS 264
7.10.4 DFSMSrmm CDS and DFSMShsm consistency checking 273
7.10.5 Using DFSMSrmm with BTLS. 273
7.10.6 Using DFSMSrmm with non-IBM libraries. 275

Chapter 8. System-managed library support 281
8.1 Storage management subsystem-managed tape library overview 282
8.2 Sharing a library between multiple systems 282
8.3 System-managed library partitioning. 283
8.3.1 Cartridge insert process 283
8.3.2 The sequence of processing. 285
8.3.3 Partitioning a library between MVS and other systems. 285
8.3.4 Partitioning a library by CBRUXENT and EDGUX200 user exits 286
8.3.5 Partitioned library with shared CDS and TCDB 286
8.3.6 Partitioned library with shared TCDB and different CDSs. 287
8.3.7 Partitioned library with unshared CDS and TCDB 287
8.3.8 Partitioned library with shared CDS and unshared TCDB 288
8.4 Partitioning support 289
8.5 Using PRTITION and OPENRULE commands. 290
8.5.1 Defining PRTITION commands. 291
8.5.2 Defining OPENRULE commands 297
8.5.3 Testing various OPENRULE settings 302
8.6 Converting REJECT commands 318
8.6.1 Examples of converting REJECT commands 319
8.7 Conversion from REJECT to PRTITION and OPENRULE 321
8.7.1 Partitioned library with shared CDS and TCDB 323
8.7.2 Partitioned library with shared TCDB and different CDSs. 324
8.7.3 Partitioned library with unshared CDS and TCDB 325
8.7.4 Partitioned library with shared CDS and unshared TCDB 327

Chapter 9. Catalog synchronization 329
9.1 Overview of catalog synchronization. 330
9.1.1 DFSMSrmm catalog processing 331
9.1.2 Reasons to re-synchronize catalog synchronization. 331
9.2 Preparing DFSMSrmm catalog synchronization 332
9.2.1 Message data set considerations 333
9.2.2 Activity log data set considerations (optional). 333
9.2.3 Journal considerations 334
9.2.4 Creating system IDs 335
9.2.5 Disabling catalog synchronization. 337
9.3 Synchronizing with a shared catalog environment 338
9.3.1 Data sets without create system ID information 340
9.3.2 Clearing the catalog synchronization 341
9.3.3 Updating the EDGRMMxx DFSMSrmm options on all systems 341
9.3.4 Synchronizing your catalogs 342
9.3.5 Checking catalog synchronization. 343
9.3.6 Enabling the journal data set (optional) 344
9.3.7 Catalog re-synchronization with a shared catalog environment 345
9.4 Synchronizing when catalogs are not shared 345

9.4.1 Setting the system creation ID if catalogs are not shared . . . 348
9.4.2 Clearing the catalog synchronization . . . 350
9.4.3 Updating the EDGRMMxx DFSMSrmm options . . . 350
9.4.4 Synchronizing your catalogs . . . 352
9.4.5 Enabling the journal data set (optional) . . . 355
9.4.6 Housekeeping considerations . . . 355
9.4.7 Catalog re-synchronization with a shared catalog environment . . . 356

Chapter 10. Enabling ISPF Data Set List (DSLIST) support . . . 357
10.1 Implementation steps . . . 358
10.2 Use the ISPF Configuration Utility . . . 358
10.3 Using the ISPF Data Set List Utility support . . . 363
10.4 Move the ISPCFIGU module to the SISPLPA library . . . 367

Chapter 11. Policy management . . . 369
11.1 VRS-related parmlib options . . . 370
11.1.1 CATRETPD . . . 371
11.1.2 GDG option . . . 372
11.1.3 MOVEBY . . . 374
11.1.4 RETAINBY . . . 375
11.1.5 REUSEBIN . . . 377
11.1.6 VRSCHANGE . . . 377
11.1.7 VRSJOBNAME . . . 378
11.1.8 VRSMIN . . . 379
11.1.9 VRSEL . . . 379
11.1.10 VRSDROP . . . 380
11.1.11 VRSRETAIN . . . 381
11.2 VRS types . . . 383
11.2.1 Data set VRS . . . 383
11.2.2 Volume VRS . . . 388
11.2.3 Name VRS . . . 388
11.3 Specifying the VRS parameters . . . 389
11.3.1 DSNAME and NAME retention types . . . 390
11.3.2 Special NAME retention type . . . 391
11.3.3 Volume retention types . . . 391
11.3.4 Retention limits . . . 391
11.3.5 Movement policies . . . 392
11.3.6 Release options . . . 393
11.4 Chaining VRSs . . . 393
11.4.1 VRS chain and subchain . . . 394
11.5 VRSEL GDG option . . . 395
11.5.1 Generation data set and DSN VRS creation . . . 399
11.5.2 CYCLEBY GENERATION and DUPLICATE BUMP . . . 401
11.5.3 CYCLEBY CRDATE and DUPLICATE BUMP . . . 402
11.5.4 CYCLEBY GENERATION and DUPLICATE BUMP with generation wrap . . . 404
11.5.5 CYCLEBY CRDATE and DUPLICATE BUMP with generation wrap . . . 406
11.5.6 CYCLEBY GENERATION and DUPLICATE DROP . . . 407
11.5.7 CYCLEBY CRDATE and DUPLICATE DROP . . . 408
11.5.8 CYCLEBY GENERATION and DUPLICATE KEEP . . . 409
11.5.9 CYCLEBY CREATION and DUPLICATE KEEP . . . 410
11.5.10 CYCLEBY GENERATION and DUPLICATE COUNT . . . 411
11.5.11 CYCLEBY CREATION and DUPLICATE COUNT . . . 412
11.6 Managing DFSMShsm tapes . . . 414

11.7 DFSMS ACS support 414
11.8 Modifying VRS 415
11.9 Deleting VRS. 415
11.10 DFSMSrmm VRS policy management simplification 416
11.10.1 Separation of the data set name mask from the policy. 417
11.10.2 Release options applied if VRS matched 418
11.10.3 Special ABEND, DELETED, and OPEN via DSNAME match 420
11.10.4 Find unused VRSs 421
11.10.5 VRS last reference date 422
11.10.6 Incomplete VRS chains and dummy VRS named *broken* 425
11.10.7 Tolerating and removing old functions 426
11.10.8 Conversion to DFSMSrmm from other tape management systems 427
11.11 Trial run 430
11.12 VRS examples 431
11.12.1 ADDVRS operand defaults 431
11.12.2 Adding a single VRS 431
11.12.3 VRS chain with one NEXTVRS operand. 432
11.12.4 VRS chain with two NEXTVRS definitions 433
11.12.5 VRS chain with two NEXTVRS operands with retention information 434
11.12.6 VRS chain with two ANDVRS operands 435
11.12.7 VRS chain with two subchains 436
11.13 Complex VRS chain with three subchains 437
11.14 VRS hints and tips. 438

Chapter 12. DFSMSrmm automatic class selection support 441
12.1 DFSMSrmm SMS ACS support 442
12.1.1 How ACS support works 443
12.1.2 ACS support for non-system-managed volumes 444
12.1.3 Non-system-managed tape libraries 445
12.1.4 SMS read-only variables. 446
12.1.5 Implementing SMS ACS processing. 451
12.1.6 Migration considerations 466
12.2 Using DFSMSrmm with virtual tape solutions 469
12.2.1 VTS support for stacked volumes 470
12.2.2 Stacked volume support for non-VTS 473

Chapter 13. System Authorization Facility tape security 475
13.1 Tape data set authorization 476
13.1.1 Suggestions for tape security 476
13.1.2 Overview of TAPEVOL and TAPEDSN processing 477
13.1.3 How the SAF tape data set authority checking works. 480
13.2 Activating RACF TAPEVOL and TAPEDSN 483
13.2.1 Before you can start 484
13.2.2 DFSMSrmm RACF tape security support 484
13.2.3 DFSMSrmm automatic tape security support 485
13.2.4 DFSMSrmm OPTION command operand TPRACF 485
13.2.5 DFSMSrmm VLPOOL command operand RACF 486
13.2.6 Running DFSMShsm and DFSMSrmm. 486
13.2.7 Defining RACF profiles 486
13.2.8 Final RACF activities. 487
13.3 Implementation SAF tape security 488
13.3.1 Check all high-level qualifiers on tape. 489
13.3.2 Update the DEVSUPxx parmlib member 489

13.4 Removing TAPEVOL and TAPEDSN processing . . . 492
13.4.1 Check and modify your RACF settings . . . 492
13.4.2 Check and modify your DFSMShsm settings . . . 495
13.4.3 Clean up your TAPEVOL profiles by using DFSMSrmm settings . . . 496
13.4.4 Clean up your TAPEVOL profiles by using commands. . . . 497
13.5 Error messages . . . 499
13.6 Testing various security settings . . . 501
13.6.1 Test case 19 . . . 539

Chapter 14. Report Generator . . . 573
14.1 Report Generator overview . . . 574
14.2 Setting up the Report Generator for your installation . . . 574
14.3 Create a report . . . 581

Chapter 15. Splitting and merging your DFSMSrmm control data set . . . 595
15.1 When to merge or split . . . 596
15.1.1 During conversion . . . 596
15.1.2 Changing the number of systems . . . 596
15.1.3 Considerations . . . 596
15.1.4 Control record . . . 597
15.1.5 Action record . . . 597
15.1.6 Data set record . . . 597
15.1.7 VRS record . . . 598
15.1.8 Owner record . . . 598
15.1.9 Product record . . . 599
15.1.10 Rack number record . . . 599
15.1.11 Bin number record . . . 601
15.1.12 Volume record. . . . 602
15.2 PARMLIB options . . . 603
15.2.1 LOCDEF . . . 603
15.2.2 VLPOOL . . . 603
15.2.3 REJECT . . . 603
15.2.4 OPENRULE . . . 604
15.2.5 PRTITION . . . 604
15.3 Running IDCAMS REPRO . . . 604
15.4 Merging the CDS. . . . 604
15.5 Splitting the CDS . . . 608
15.5.1 Converting to an existing CDS . . . 613
15.6 Merging DFSMSrmm environments . . . 614
15.6.1 One or more RMMplexes . . . 614
15.6.2 Considerations for merging RMMplexes . . . 614
15.6.3 CDS record types . . . 616
15.6.4 Step 3: Review the DFSMSrmm PARMLIB options . . . 622
15.6.5 Step 4: Review the DFSMSrmm started task JCL . . . 623
15.6.6 Methodology for merging RMMplexes . . . 625
15.6.7 Tools and documentation . . . 628

Appendix A. Security topics . . . 633
RACF implementation . . . 634
Assigning a RACF user ID for DFSMSrmm . . . 634
Identifying DFSMSrmm to RACF . . . 634
Defining DFSMSrmm resources to RACF. . . . 636
Defining RACF groups for DFSMSrmm users . . . 638
CA-Top Secret implementation . . . 656

Defining user groups 656
Protecting DFSMSrmm data sets 657
Defining DFSMSrmm resources 658
ACF2 implementation 659
Defining data set rules 660
Defining DFSMSrmm resources 661

Glossary 665

Related publications 685
IBM Redbooks 685
Other publications 685
Online resources 686
How to get Redbooks 686
Help from IBM 686

Index 687

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

FICON®
HyperSwap®
IBM®
Magstar®
MVS™
OS/390®
Parallel Sysplex®
RACF®
Redbooks®
Redbooks (logo) ®
Resource Measurement Facility™
RMF™
System Storage®
System z®
System/390®
Tivoli®
VTF®
WebSphere®
z/OS®

The following terms are trademarks of other companies:

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Preface

DFSMSrmm from IBM® is the full function tape management system available in IBM OS/390® and IBM z/OS®. With DFSMSrmm, you can manage all types of tape media at the shelf, volume, and data set level, simplifying the tasks of your tape librarian.

Are you a new DFSMSrmm user? Then, this IBM Redbooks® publication introduces you to the DFSMSrmm basic concepts and functions. You learn how to manage your tape environment by implementing the DFSMSrmm management policies.

Are you already using DFSMSrmm? In that case, this publication provides the most up-to-date information about the new functions and enhancements introduced with the latest release of DFSMSrmm. You will find useful information for implementing these new functions and getting more benefits from DFSMSrmm.

Do you want to test DFSMSrmm functions? If you are using another tape management system and are thinking about converting to DFSMSrmm, you can start DFSMSrmm and run it in parallel with your current system for testing purposes.

This book is intended to be a starting point for new professionals and a handbook for using the basic DFSMSrmm functions.

Authors

This Redbooks publication was produced by a team of IBM specialists from around the world working at the International Technical Support Organization (ITSO), San Jose Center.

Mary Lovelace is a Consulting IT Specialist at the International Technical Support Organization. She has more than 25 years of experience with IBM in large systems, storage, and storage networking product education, system engineering and consultancy, and systems support. She has written many IBM Redbooks publications about z/OS storage products, IBM Tivoli® Storage Productivity Center, Tivoli Storage Manager, and Scale Out NAS.

Norbert Schlumberger is an IT Architect with IBM Germany. He has 35 years of experience in storage software and storage management for IBM and client systems, including 24 years of experience with DFSMSrmm. He has developed, delivered, and taught DFSMSrmm education to clients around the world. He has many tools available, such as a Tape Copy Tool to support the DFSMSrmm business. His areas of expertise include performing conversions from vendor tape management products to DFSMSrmm and new DFSMSrmm implementations. Norbert provides marketing support for DFSMSrmm, including IBM 3494 and IBM 3495 Automated Tape Libraries (ATLs), Virtual Tape Servers (VTSs), and TS7700 Virtual Engines. He has worked at IBM for 39 years.

The original authors of this book are listed:

Yolanda Cascajo Jiménez
Sue Hamner
Andreas Henicke
Begoña Fernández Vila
Dong Rui Ling

Thanks to the following people for their contributions to this project:

Robert Haimowitz
ITSO, Raleigh Center

Vickie Dault
IBM USA

Thomas Berlinghof
Rolf Hallek
Klaus Jaeger
Peter Pietzsch
Uwe Roettenbacher
Michael Welsch
Wolf Wittenbecherr
IBM Germany

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. QXXE Building 80-E2
  650 Harry Road
  San Jose, California 95120-6099

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

DFSMSrmm enhancements

You might be an experienced Data Facility System Managed Storage removable media manager (DFSMSrmm) user who is thinking of migrating from your current DFSMSrmm release up to z/OS V1R13. If so, you will find the information in this chapter useful. This chapter documents the new functions that are included in z/OS V1R3 up to z/OS V1R13, what users will find different as compared to the previous release, and what is needed to use each new function. We refer to related chapters for details about each function.

The new function of the following z/OS DFSMSrmm releases is described:

- “Changes introduced with z/OS V1R13 DFSMSrmm” on page 2
- “Changes introduced with z/OS V1R12 DFSMSrmm” on page 4
- “Changes introduced with z/OS V1R11 DFSMSrmm” on page 5
- “Changes introduced with z/OS V1R10 DFSMSrmm” on page 8
- “Changes introduced with z/OS V1R9 DFSMSrmm” on page 12
- “Changes introduced with z/OS V1R8 DFSMSrmm” on page 15
- “Changes introduced with z/OS V1R7 DFSMSrmm” on page 16
- “Changes introduced with z/OS V1R6 DFSMSrmm” on page 17
- “Changes introduced with z/OS V1R5 DFSMSrmm” on page 18
- “Changes introduced with z/OS V1R4 DFSMSrmm” on page 19
- “Changes introduced with z/OS V1R3 DFSMSrmm” on page 20

1.1 Changes introduced with z/OS V1R13 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R13.

1.1.1 Customize point-and-shoot fields in the DFSMSrmm dialog

To easily see the fields that are enabled for point-and-shoot, you must customize the color, intensity, and highlighting of the point-and-shoot fields. Issue the ISPF system command PSCOLOR from any ISPF command line and adjust the Point-and-Shoot Panel Element.

1.1.2 Specifying the retention method to use for new tape data sets

Among the important decisions to be made when using DFSMSrmm is how to retain tape data sets and for how long. You might want to retain a given data set for a specific period of time after it is created, or retain it based on some event (for example, while the data set is cataloged), or retain it permanently.

DFSMSrmm provides two retention methods for retaining tape data sets:

- The EXPDT retention method, which allows expiration to be on a specific date. This is the expiration date.
- The VRSEL retention method, which uses vital record specification (VRS) to implement retention and movement policies through which a retention date is calculated each time that VRSEL inventory management processing is run. DFSMSrmm retains a volume based on this retention date and on the volume expiration date.

You can use the EDG_EXIT100 installation exit to set the retention method to be used for new tape data sets. When you create a new tape volume set, or rewrite an existing set from the first file, you can override the system default retention method.

Note: Use the RETENTIONMETHOD parmlib option if you want to set a default system-wide retention method. If you do not specify a default retention method, the system uses the VRSEL retention method as the default.

1.1.3 IBM TS1140 support

DFSMSrmm now supports the new media types MEDIA11, MEDIA12, and MEDIA13, as well as the new recording formats EFMT4 and EEFMT4. The following list provides the correspondence between the DFSMSrmm media types and the media types used in the system-managed tape data class construct:

MEDIA11/EATC IBM Enterprise Advanced Tape Cartridge
MEDIA12/EAWTC IBM Enterprise Advanced WORM Tape Cartridge
MEDIA13/EAETC IBM Enterprise Advanced Economy Tape Cartridge

1.1.4 Copying data set attributes in a tape copy application

A tape copy application can use the EDG_EXIT100 installation exit to copy the data set attributes from the original to the copy during OPEN processing. In the EDG_EXIT100 installation exit, you can specify the data set details for the source data set from which attributes are to be copied.

1.1.5 Excluding specific data sets from VRSEL processing

You can use the EDG_EXIT100 installation exit to exclude specific data sets from DFSMSrmm VRSEL processing as they are created or rewritten. You can specify this for any data set, but DFSMSrmm ignores the request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes that are managed by the EXPDT retention method and it is not affected by this support.

When DFSMSrmm excludes a data set from VRSEL processing, it ensures that the data set vital record attribute is reset and the retention date is set to the current date. The matching VRS information is left unchanged.

1.1.6 Inventory management considerations for EXPROC and VRSEL

The inventory processing for processing vital records and performing expiration processing has been changed:

- You do not need to run VRSEL processing unless volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle expiration of all volumes managed by the EXPDT retention method.
- EXPROC processing provides a summary of volumes by retention method.
- The expiration date of volumes is set during OPEN processing, so for volumes managed by the EXPDT retention method, no special considerations exist for open data sets; they are managed based on the volume EXPDT.
- For volumes that are managed by the EXPDT retention method, no special considerations exist for data sets closed by ABEND processing or that are DELETED; they are managed based on the volume EXPDT.
- Volumes that are managed by the EXPDT retention method are included only in the EXPDTDROP limit. VRSRETAIN and VRSDROP limits apply only to volumes that are managed by the VRSEL retention method.

1.1.7 Volume usage, capacity, and compression considerations

DFSMSrmm tracks information about what you write onto tape volumes, including record length, block size, and number of blocks. DFSMSrmm also tracks information about the data written to the tape volume, such as the compressed size and physical size of the file and can calculate the compression ratio. In addition, information is recorded about the media capacity, percentage used, physical space used, and the overall compression ratio.

Basic information about the media can be defined by command, but, when a volume is used for output, DFSMSrmm records the media type, recording format, capacity, and percentage used. Normally, this information is provided by the tape drive when the drive is an IBM 3590, or IBM TS11x0 or later drive. For older tape devices, such as 3490, and also for virtual tape emulating 3490, this information is not available and is derived from media information hard coded in DFSMSrmm. For non-IBM media, and to override these details for IBM media, you can define media information to DFSMSrmm using the MEDINF command in the DFSMSrmm parmlib.

1.2 Changes introduced with z/OS V1R12 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R12.

DFSMSrmm provides multiple enhancements in z/OS V1.12. The ACTIVITY file is updated to include the reason why a DFSMSrmm retention limit was reached. This function is also available now for z/OS V1.10 and z/OS V1.11 with the PTF for APAR OA30881. New reports created from the ACTIVITY and extract files are intended to help you see why retention limits were triggered. Also, OPENRULE ignore processing is available for duplicate tape volumes. Support is intended to allow you to set a volume hold attribute to prevent expiration and to search and report on volumes that have the hold attribute. It is also intended that the DFSMSrmm ISPF dialog search results can be bypassed when using the CLIST option. The following enhancements are new:

- Additional z/OS V1.12 scalability and performance improvements

  DFSMS now supports additional data set types, including basic and large format sequential data sets, partitioned (PDS/PDSE) data sets, direct (BDAM) data sets, and catalogs in the extended addressing space (EAS) on extended address volumes (EAVs). Support is also included for generation data groups (GDGs) and Virtual Storage Access Method (VSAM) volume data sets (VVDSs). Overall, EAV helps you to relieve storage constraints, as well as to simplify storage management by providing the ability to manage fewer, large volumes as opposed to many small volumes.

  Another new EAV-related function is added support to make all data sets used by DFSMSrmm eligible for allocation in the extended addressing space of an EAV. This includes the DFSMSrmm journal and dynamically allocated temporary files.
- With z/OS V1.12 DFSORT, information about DFSMSrmm active and queued tasks is available via the DFSMSrmm API and via a TSO/E subcommand, enabling storage applications to monitor and act on the available information. In addition, you can use the DFSMSrmm dialog to manage these tasks.
- Details about the optimization improvements for z/OS V1.12

  DFSMSrmm helps with the reporting of data sets and logical volumes that are copy-exported from a TS7700 Virtualization Engine.
- Integrated Storage Management Facility (ISMF) includes a Data Collection application, DCOLLECT, which provides storage-related measurement data that can be used as input to the DFSMSrmm Report Generator to create customized reports or to feed other applications, such as billing applications. In z/OS V1.12, DCOLLECT data class (DC) records are updated to include information about all data class attributes. Also, data set (D) records now include job names, and storage group (SG) records now include information about Object Access Method (OAM) Protect Retention and Protect Deletion settings.
- In z/OS 1.12, DFSMSrmm supports IPv6.

1.2.1 Ease of use and flexibility enhanced

- Retention limit reporting
- Ignore for duplicate volumes
- Automation for WTORs in production and parallel running
- Expiration override for volumes
- ISPF dialog CLIST option to avoid search results list

1.2.2 Simplification

Now, you have the availability to request information about DFSMSrmm subsystem address space status, tasks, and queued requests.

In addition to the MODIFY operator command, you can now use the RMM LC STATUS subcommand to retrieve information about the DFSMSrmm subsystem, subsystem requests, and task status.

1.2.3 Storage reporting

EDGJCEXP provides a report about copies of logical volumes that are exported from TS7700 Virtualization Engine. The report consolidates point-in-time information from the copy export status file, library, and DFSMSrmm to help you identify tape data that was copy-exported.

1.2.4 Availability

The DFRMM shutdown now issues an additional message to list the job names of the address spaces preventing shutdown. The DFSMSrmm subsystem interface processing now correctly detects that DFRMM is or has been stopped.

With DFSMSrmm on z/OS V1R12 and later releases, the ASID/JOB value is automatically added to all trace records that are created outside of the DFRMM subsystem address space. This enables the address space identifier (ASID) and jobname to be stored to the trace record.

1.2.5 Scalability and performance

- IPV6 supports networking strategy
- EAV supported for all data sets
- External authorization service (EAS) exploited for temporary data sets
- Extended task input/output table (XTIOT), data set association block (DSAB) above, and uncaptured unit control block (UCB) support

1.3 Changes introduced with z/OS V1R11 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R11.

1.3.1 EDGINERS tape label scan

The DFSMSrmm tape utility EDGINERS performs initialization and erasure of tapes. For tape label read and display, another utility, such as DITTO or File Manager, must be used. Now, EDGINERS is updated to support the reading and cross verification of tape label information with the records defined in the DFSMSrmm control data set. The new function SCAN helps you with identifying and managing tapes that come from other systems or are in a problem state. The EDGINERS SCAN function reads the VOL1 and header labels for the first file on the specify volume.

If you have READ authorization, you can SCAN these volumes:

- Storage management subsystem (SMS) managed or not SMS managed
- Known to Removable Media Manager (RMM) or not known to RMM

- Of all status values except damaged volumes
- Of all types, including WORM and encrypted tapes

1.3.2 Dynamic installation exits

DFSMSrmm now uses the system Dynamic Exit Services to manage calls to the installation exits, determine whether exit modules exist, and to provide error handling and recovery. The installation exits DFSMSrmm provides are shown in Table 1-1.

Table 1-1 Sample installation exits IBM provides

Exit name	Default exit routine	SAMPLIB samples provided
EDG_EXIT100	EDGUX100	EDGUX100 and EDGCVRSX
EDG_EXIT200	EDGUX200	EDGUX200
EDG_EXIT300	EDGUX300	EDGUX300

DFSMSrmm installation exit modules can be loaded from any authorized program facility (APF)-authorized library. The default is the LINKLST.

Dynamic Exit Services is used to load and activate the default (EDGUXn00) exit module at initialization time. The default exit is activated in first position. All other exit-related processing is handled by Dynamic Exit Services. If you do not use the default exit module (EDGUXn00), you must use PROGxx in the z/OS parmlib or the SETPROG operator command, or the CSVDYNEX macro.

1.3.3 Returning volumes to the system-managed library

The sample volume that is not in the library installation exit, CBRUXVNL, is enhanced to enable fewer installations to require customization. It is intended that volumes will be requested to be entered into the tape library whenever possible.

Additional checks and customization are added to the sample volume not in the library installation exit so that there are more cases where a decision can be made. **EDG8121D** is now issued if a tape configuration database (TCDB) volume entry exists. Or, EDG8121D is now issued if the volume is known to be in a different library, but only if the volume is defined to DFSMSrmm, the volume is not on loan, and the volume will not be rejected during OPEN (status is pending release, init action pending, volume is scratch, or not for use on IBM MVS™).

When you regularly store system-managed volumes outside of a system-managed library, and also have the home location set to SHELF (or some other LOCDEF-defined storage location with a type of HOME), you can ensure that the volumes will be called for by CBRUXVNL by using the SMSTAPE(PURGE(NO)) option. This results in the TCDB volume entry being retained while the volume is ejected. The DFSMSrmm-supplied sample exit checks whether the TCDB entry exists and, for DFSMSrmm-managed volumes, will prompt the operator to re-enter the volume into a library.

An alternative to the SMSTAPE(PURGE(NO)) option is to use SMSTAPE(PURGE(ASIS)) and ensure that the library eject default is set to KEEP. This enables you to be selective about the libraries to which the purge/keep option applies. SMSTAPE(PURGE(ASIS)) is the default.

1.3.4 VRSEL GDG option

New DFSMSrmm parmlib options provide flexibility in how tape generation data sets are managed for cyclic retention.

Use the GDG option to specify how generation data groups are handled for cycle retention by VRSEL processing. Cycle retention includes both the CYCLES and the BYDAYSCYCLE retention types. The correct sequence for determining the retention can be either using the generation number or using the creation order. You can also specify how duplicate generations are handled and have the flexibility to include or exclude duplicates from the cycles count as required by your application processing.

1.3.5 DFSMSrmm Report Generator

Extensive changes are made for the DFSMSrmm Report Generator, improving usability, enabling more customization of reports, and simplifying the way that selection information can be specified. This is accomplished through exploitation of recent changes to DFSORT and ICETOOL, data typing, and report type inheritance. The changes further improve the reporting that is available for DFSMSrmm, DFSMShsm, and other DFSMS components. The following list explains the changes:

- You can now override a data type within the dialog. The updated data type is saved in the report type and report definition. The Report Generator remembers the original data type and your override. You can use report type inheritance to benefit from any new data type values in report types.
- You can now inherit changes in report types into existing report definitions. This enables changes in data types, comments, and other criteria to be merged into pre-existing report definitions. This improves usability and enables your reports to benefit from improved report types shipped with DFSMSrmm.
- You can now select which fields are not to be included in totals and break totals. When you specify that a field used for a report column is not to be subject to totaling, the Report Generator uses the NOST option with the ICETOOL reporting tool. When used with ICETOOL, all numeric fields are automatically totaled unless you request that they are excluded.
- You can now specify a list of field values and the text to be used for them in the report. The Report Generator uses existing field equate values to construct an initial list of possible values; before you can use any of these values, you must provide a new value to which the field will be changed for the report. Only those which have a change value are used for report generation.
- When you specify the column width to be used, the generated ICETOOL statements now override the default ICETOOL processing. If no override is provided, the default is to set the width at a maximum of the column header text, and the data size. The data size is listed:
  - Ten characters for 4-byte fields, five characters for 2-byte fields, and three characters for 1-byte fields of binary data.
  - For numeric fields declared with Z (zoned) or P (packed), the default is set to the number of possible digits.
- You can now use the equated assembler symbols instead of the absolute value. An option in the dialog displays the available equates that are ready for you to use. When equates are available, the Report Generator uses these as a basic set to enable support for the specification of an alternate value for use in the report – this is called a "*change value*". For example, "I" can be changed to "INFO" and so on.

- Existing report types and definitions are updated with help and guidance information, such as what variables to set in JCL, how to run HSM preprocessor to convert data, and so on. This information is presented to the user on request and when generating JCL. The help information is split into three parts; Type, Report, and JCL help. It is browsed or edited as a single set of help information. You can edit and add to this information, which is stored in the definitions.
- You can now request that the reports created from the DFSMSrmm report extract include the date and time when the extract was created. The report extract type and samples are updated to exploit this information. In fact, you can make this request for any type of report where the input records include one or more values that you want to include in the report title. This exploits the new DFSORT ICETOOL capability to specify multiple report TITLE strings.
- A new reporting tool is shipped that enables records to be manipulated with DFSORT. The output of this tool is not a report, but rather reformatted records. When using the new reporting tool, the Report Generator can be considered as a partial replacement for the now withdrawn DFSORT ISPF panels. The JCL generated by the reporting tool includes comments that contain DFSORT symbol definitions so that you can easily process the record further using DFSORT or ICETOOL.
- All report types are updated to include the relevant data types and help information. In addition, all report samples are updated to inherit all new information from the report type, including the data type and help information.
- You can now use substrings for record selection criteria regardless of the data type of the field.

1.3.6 DFSMSrmm usability items

DFSMSrmm includes the following usability items:

- Use of MATCHVRS in the ISPF data set dialog
- New ADDVOLUME subcommand fields
- Extended SEARCHVOLUME subcommand
- VRS location definition handling
- REXX variables
- DFSMSrmm journaling
- Using the EDGUPDT utility
- API multi-entry return

1.3.7 VRSEL(OLD) parmlib option

You cannot use the VRSEL(OLD) parmlib option. Before the z/OS V1.11 release, EDGHSKP issued a warning message when VRSEL(OLD) was in use.

1.4 Changes introduced with z/OS V1R10 DFSMSrmm

This section describes the enhancements to DFSMSrmm that are introduced with z/OS V1R10.

1.4.1 Enable use of DFSMSdss COPY Service

With DFSMSrmm release z/OS 1.10, you can request that DFSMSrmm back up the control data set and the journal, using the DFSMSdss COPY command.

DFSMSrmm no longer relies on concurrent copy and virtual concurrent copy for fast and non-intrusive creation of copies and backups of the DFSMSrmm control data set. DFSMSrmm supports the use of Data Set Services (DSS) copy services and exploitation of Fast Replication services that are provided by DASD subsystems. This change enables almost instantaneous copies of the control data set to be created. This enables reduced recovery time and eliminates the impact that the use of concurrent copy can have on IBM HyperSwap®.

DFSMSrmm control data set backup can now additionally be performed using copy services. ADRDSSU is used with Fast Replication to enable a control data set “backup” to be created. New parameter options are provided to request this method for the DFSMSrmm control data set backup.

1.4.2 SMF record changes

RMM creates SMF records when requested, but can now optionally use the IBM assigned record type 42 instead of using SMF record types from the user-range. SMFAUD and SMFSEC options each use a record type unless the IBM record type is used, in which case subtypes are used within record type 42. The subtypes used by DFSMSrmm are 22 for audit records and 23 for security records.

1.4.3 Audit controls for release processing

The existing parmlib options for ensuring that VRS-managed retention is as expected include the VRSMIN and VRSCHANGE operands. If you want to control which volumes are released, you have to exploit the NOTIFY release option of VLPOOL so that all volumes can be kept in pending release status until the list of volumes is validated against some installation-based criteria. The latter is a manual action that is required, but it can easily be automated.

New parmlib options are added, which can be used to ensure that data is being retained as expected and only released by DFSMSrmm processing if the numbers or percentages of volumes are within policy limits. EDGHSKP VRSEL and EXPROC processing counts the numbers of volumes in the following categories:

- VRS retained; newly and initial count
- Newly assigned since the last VRSEL run
- Planned to be set pending release; dropped from VRS and dropped by EXPDT
- EXPDT retained initial count

It applies the new parmlib options to determine the action to be taken. New messages are added to the MESSAGE file, and are issued if VRSEL or EXPROC are run and the corresponding parmlib option has not disabled the function. The new messages list the numbers and percentages of volumes. Regardless of disablement, some existing messages have a percentage added for completeness. In addition, based on the set limit and action, an existing message EDG2310I is issued if the action is FAIL, and the EDGHSKP return code is set if the limit threshold is exceeded. This processing for the new options is identical to the existing processing for VRSMIN.

1.4.4 XREPTEXT tailoring via SYSIN

You can create an extract data set to use as input for creating reports and select the records to be extracted from the DFSMSrmm control data set. Specify the RPTEXT command in the EDGHSKP SYSIN file to select the records that you want to be written to the report extract file. When you specify the XREPTEXT DD statement, the extract contains only extended records that are a combination of data set information and volume information that can be useful as input to DFSMSrmm reporting tools.

Report extract processing now checks for the existence of the RPTEXT command in the SYSIN file. If the RPTEXT command is found, it writes the selected extract records to either the REPTEXT or XREPTEXT DD. XREPTEXT DD is checked for and used in preference. When you code only the REPTEXT DD, all records other than the extended records are created.

When there is no RPTEXT command in SYSIN or the RPTEXT command specifies no RECORDS operand, the default is that the records written depend on the extract file DD being used:

REPTEXT RMM creates all records except extended records in the extract (unchanged).

XREPTEXT RMM creates only extended records (changed).

The EDG6013I message is used to list the RPTEXT command options from SYSIN. This is done in the same way that EXPROC is already listed.

1.4.5 Volume replacement policies

New limits and policies are used to determine whether to set the REPLACE actions. MEDINF commands in parmlib can now include one REPLACE policy setting for each media type and recording format combination.

The place and timing for the checks for the limits is unchanged. In addition, these new functions are available:

- DFSMSrmm now checks at OPEN time to see whether a volume to be used for output processing has the REPLACE action pending and rejects the volume. If the REPLACE action is pending, the volume must be pending release and already subject to reject by DFSMSrmm Open/Close/End of Volume (O/C/EOV) processing.
- REPLACE release action is set, an owner is set for the volume, and the volume is set to pending release.
- DFSMSrmm intercepts WTO messages that contain SARS MIM information and sets the REPLACE release action for any volume where the suggested recovery action is to copy off the data or replace the volume. See WTO IEA486E. In addition, for scratch volumes, an owner is set for the volume and the volume is set to pending release.

1.4.6 Report Generator enhancements

The DFSMSrmm Report Generator is an Interactive System Productivity Facility (ISPF) application that you can use to create reports. The Report Generator offers these functions:

- Provides reports that you can run as-is or that you can modify as you want. You can use samples to create reports for volumes, data sets, racks, owners, and the retention and movement policies that are established for your installation. You can modify these samples

to create tailored reports. DFSMSrmm ships samples in `SYS1.SAMPLIB`. See Chapter 14, “Report Generator” on page 573 to run one of these reports.

- Generates job control language (JCL) that is based on specifications that you use to submit the report jobs. The generation of JCL depends on the report type and therefore the macros that map the data records. The generation knows, based on the macro name and keyword options used, whether to generate a DCOLLECT job step, a DFSMShsm FSR reformat, a DFSMSrmm extract, or a copy of SMF records.
- Includes samples for reporting from DCOLLECT and DFSMShsm data.

The DFSMSrmm Report Generator is updated to support keywords for assembler macros from which report types are derived, and to add new built-in data extract steps. This is in support of new SMF record types from DFSMSrmm, and for DFSMShsm and DCOLLECT reporting.

Any new report type or report definitions created on z/OS V1R10, which include macro keyword information, can be used by a lower-level release. However, the macro keywords are ignored and are removed if the report type or report definition is changed or updated. The original input report type or report definition is unaffected and can be reused later on a supporting release to exploit the macro keywords.

1.4.7 System-managed library partitioning

In z/OS 1.10, new parmlib controls are provided to enable simplified and more powerful partitioning of system-managed libraries and control of the application use of tape volumes. Now, you can use PRTITION and OPENRULE commands in the EDGRMM parmlib member to control library partitioning and the use of volumes by applications.

Note: If you use REJECT commands, you have to convert from the use of REJECT commands to use the PRTITION and OPENRULE commands.

You can partition a system-managed library, including a Virtual Tape Server (VTS), by performing these tasks:

- Specify the USE operand value on the RMM ADDVOLUME or RMM CHANGEVOLUME subcommands. You can set this value to MVS, VM, or both. If you do not specify MVS for a volume, DFSMSrmm prevents the volume from being defined in the volume catalog on this system.
- Through the use of the PRTITION and OPENRULE parmlib commands, you can simplify the maintenance of the parmlib members as your libraries and volume ranges change. Operands on the OPENRULE and PRTITION commands allow global actions to be set. You can use one or more specific overrides based on volume sets that have different requirements. Typically, you can add a new range of volumes for use by a single partition and only that one system needs to be updated. The OPENRULE and PRTITION commands allow you to define whether they apply to volumes defined to DFSMSrmm or not. You can use operands on the OPENRULE command to automatically ignore volumes, rather than using EXPDT=98000, ACCODE=xCANORES, or a customized EDGUX100.
- Define parmlib member EDGRMMxx REJECT prefixes. You can use REJECT to prevent a volume not defined to DFSMSrmm from being defined in a system-managed tape library. The REJECT ANYUSE(prefix) operand prevents a volume from being defined in the system-managed tape library on the current system. The REJECT OUTPUT(prefix) operand allows you to define the volume to the system-managed tape library but only use the volume for input processing.

1.4.8 Common Information Model (CIM) Provider

The DFSMSrmm CIM agent now has an option to register itself using the Storage Management Initiative Specification (SMI-S) Storage Library profile so that storage management clients (and Transaction Processing Council (TPC) in particular) can use Service Location Protocol (SLP) to detect the CIM agent and determine which registered profiles it can support.

DFSMSrmm now ships the web service for installation either as an enterprise archive (EAR) under IBM WebSphere® or as a Web Archive under Tomcat (WAR) or other web service middleware. For details of externals, handling, installation, and customization, see *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405.

1.5 Changes introduced with z/OS V1R9 DFSMSrmm

This section describes the enhancements to DFSMSrmm that are introduced with z/OS V1R9.

1.5.1 Task management support

Any task in the system that requests DFSMSrmm subsystem services and fails, or is interrupted because a TSO-user used Attention (ATTN), or is canceled by the operator, results in any corresponding long-running subsystem request failing. In addition, there are checkpoints built into long-running requests so that when the requestor ends (such as a job being canceled), DFSMSrmm processing is interrupted at a safe and convenient point. Long-running local tasks are DFSMSrmm subsystem requests that last long enough to be included in the results of a QUERY ACTIVE command, and the task token can be obtained and used.

1.5.2 Multitasking of utilities

The way in which DFSMSrmm utilities, EDGUTIL and EDGHSKP, interact with system-managed volumes in an IBM system-managed library is improved through multiple changes that might, especially in larger VTS installations, result in shorter elapsed time and more flexibility. The following items are discussed:

- EDGHSKP EXPROC
- EDGUTIL
- Using EDGSPLCS

1.5.3 Control data set (CDS) serialization

DFSMSrmm now requires you to supply a control data set identifier in the CSDID parameter of SYS1.PARMLIB member EDGRMMxx, specifying the identifier of the CDS to be used on that system. DFSMSrmm uses the CSDID as part of the enquiry character (ENQ) name used to serialize updates to the RMM CDS. The DFSMSrmm CIM provider also uses the CSDID to distinguish between multiple control data sets. If the CSDID is not yet set in the CDS, you can optionally use EDGUTIL UPDATE to set it; otherwise, DFSMSrmm will set the CSDID when first started. We suggest that you use a unique CSDID for each CDS.

At DFSMSrmm startup time, DFSMSrmm matches the CDSID in the control data set control record value with the CDSID operand in parmlib member EDGRMMxx.

1.5.4 JCL data set names

The data set naming requirements for tape data sets supported by the TSO subcommands and API are relaxed so that any 44-character string can be used. This enables the use of unqualified data set names, such as those that are accepted by the MVS JCL DSNAME keyword. Quoted data set name strings allow any character string to be specified for a data set name except for leading blank and leading hex zero, which are not supported.

Important: *Quotes make the difference.*

1.5.5 Shared parmlib support

Some information in the EDGRMMxx parmlib might need to be specific to a subset of your systems. For example, the REJECT or VLPOOL entries might need to be different across systems. To enable this information to be handled on a system-by-system basis, you can specify a second parmlib member to be used.

Use the MEMBER operand in the primary DFSMSrmm parmlib to identify a second parmlib member that contains overriding or additional parmlib options. Some information in the EDGRMMxx parmlib member might need to be specific to a subset of your systems.

1.5.6 TSO subcommands

TSO subcommand parsing rules are further relaxed to support different product versions and the de-classification of data sets and volumes. The following topics are discussed:

- Changes in subcommand parsing:
  - The rules for the Product Level have changed.
  - A new NOSECLEVEL parameter was introduced to reset the SECLEVEL for a volume or data set.
- CLIST enhancements:
  - New parameters START and ADD were introduced to enable the user to either overwrite or add to an existing CLIST data set.
  - Existing or user-specified DCB record format parameters are now supported.
- Search command enhancements:
  - A new CONTINUE parameter was introduced for all search commands as a way to break down search results into manageable quantities.
  - There is a new STORAGEGROUP parameter for the Search Volume command.
- Report 17

  A new report, REPORT 17, is added to the EDGRRPTE exec. It summarizes information for logical and stacked volumes to support stacked volume management.

1.5.7 3592 Model E05 software support

Using the existing media types (MEDIA5, MEDIA6, MEDIA7, and MEDIA8) and the two extended length future media types (MEDIA9 and MEDIA10), an encryption-enabled 3592 Model E05 reads and writes with the new Enterprise Encrypted Format 2 (EEFMT2) recording technology. It can also read and write using the Enterprise Format 1 (EFMT1) and Enterprise Format 2 (EFMT2) non-encrypted recording technologies.

In order to request EEFMT2 in the stand-alone and in the system-managed IBM tape library environment, a DFSMS data class must be used, which specifies EEFMT2 as its recording technology. Otherwise, EFMT2 is the default recording technology that is used. Data class can also be used to request the EFMT1 recording format and to explicitly request the EFMT2 format. A mix of recording formats is not supported on the same tape cartridge. An enhanced 3592 Model E05 that does not have the encryption feature enabled can only read and write in the non-encryption formats (EFMT1 and EFMT2), which is the same as the base 3592 Model E05. In a mixed 3592 environment, new microcode is also required for the 3592 Model J and the base 3592 Model E05 so that they recognize a volume with the new EEFMT2 recording technology.

1.5.8 REPORT17 of EDGRRPTE REXX Exec

When a stacked volume, containing exported logical volumes, is ejected from the library, as the logical volumes expire, RMM places the volumes in a “pending release” state. Then, when the logical volumes are imported into the library, RMM completes the return to scratch process enabling the volumes to be reused. As the exported logical volumes expire, you need the ability to do offsite stacked volume management so you can determine when to bring a stacked volume back onsite for possible reuse. RMM has enough information in its database for you to create and run reports. However, a specific stacked volume management report did not exist.

With z/OS V1R9, RMM provides a new stacked volume management report for clients to customize and run. It includes the ability to report on the percentage of active data on a stacked volume and to also report on the percentage of active logical volumes on a stacked volume. A new report, REPORT17, is added to the EDGRRPTE reporting exec, which summarizes information for logical and stacked volumes. The stacked volumes are presented in the order of increasing percentage of the active number of volumes and percentage used. The least used stacked volumes are listed first.

1.5.9 Common Information Model (CIM) provider

The RMM CIM provider is the link for a customer to obtain real-time RMM data within a Common Information Model (CIM) environment. This functionality will enable a CIM client, such as a PC, to obtain RMM data.

In z/OS V1R8, DFSMSrmm shipped a Common Information Model (CIM) provider for use with the OpenPegasus CIM object model (CIMOM). The provider supported a subset of the resources managed by DFSMSrmm. The CIM model supported by DFSMSrmm is further extended in this release to cover all of the resources managed by DFSMSrmm. The CIM provider is updated to support out-of-process mode functionality added to OpenPegasus CIM Server with 2.5.1. This has an impact on the way that the DFSMSrmm provider code is installed and run.

In addition, the RMM extensions to the CIM object model are based on the latest CIM V2.11 specification. To further enable selection of resources via the API, the SEARCH subcommands are all now capable of specifying continuation, enabling “chunking” of returned entries.

1.6 Changes introduced with z/OS V1R8 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R8.

1.6.1 Support for a true email address for the RMM NOTIFY function

The established DFSMSrmm notification feature is extended to optionally send mail to Internet addresses. Email notification is invoked in the same way as the known notifications. There is no difference to the current processing, only the result (email instead of the established messages) is different.

Email notification has the following prerequisites:

- A Simple Mail Transfer Protocol (SMTP) server must be available in the customer environment and known to DFSMSrmm.
- The owner attribute, “email address,” is set.

1.6.2 Setting up DFSMSrmm common time support

Before DFSMSrmm common time or Coordinated Universal Time (UTC) support is enabled, all dates and times are stored in the DFSMSrmm control data set in local time. When the control data set is shared, and the sharing systems are set to run in different time zones, the local dates and times in the control data set can be from any of your systems. When you display information or extract records, you need to be aware of how the records were created, on which system, and where they might have been updated to interpret the dates and times that are shown. The same consideration also applies for records created or updated before enabling common time support because DFSMSrmm assumes they are times local to the system running the DFSMSrmm subsystem and converts the values based on that assumption.

When you enable common time support, DFSMSrmm maintains the records in the control data set in common time. Most date and time fields are paired together to enable an accurate conversion to and from common time and between different time zones. In some cases, DFSMSrmm has date fields in control data set records, and there is no associated time field. For these date fields, DFSMSrmm uses an internal algorithm that approximates conversion between time zones based on the time zone offsets involved.

1.6.3 DFSMSrmm VRS policy management simplification

In this new release, the VRS processing changed. The RMM TSO subcommand and the reporting are enhanced:

- Separation of the data set name mask from the policy
- Release options applied if VRS matched
- Special ABEND, DELETED, and OPEN via DSNAME match
- Find unused VRSs
- Incomplete VRS chains and dummy VRS named *broken*
- Tolerating and removing old functions.
- Conversion to DFSMSrmm from other tape management systems

1.6.4 DFSMSrmm usability items

The new release has the following usability updates:

- Updates to the RMM TSO SEARCHVOLUME subcommand
- ISPF lists show retention information
- SELECT primary command in RMM dialog search results
- RMM TSO CHANGEVRS subcommand
- RMM TSO SEARCHOWNER subcommand
- REXX variable constraint relief
- Enabling ISPF data set list (DSLIST) support

1.6.5 Tape data set authorization

DFSMS V1.8 provides new options for securing tape data sets using the system authorization facility (SAF) to allow you to protect data sets on tape using the RACF DATASET class without the need to activate the TAPEDSN option or the TAPEVOL class. This new tape data set authorization checking allows you to specify that all data sets on a tape volume need to have common authorization. You can specify whether users are authorized to overwrite existing files on a tape volume. In addition, you can specify that a user must have access to the first file on a tape volume to add additional files on that tape volume.

1.7 Changes introduced with z/OS V1R7 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R7.

1.7.1 Issue DFSMSrmm TSO commands from the console

The MVS modify command can now be used to issue an RMM TSO command from the console. The command output is returned to the console and system log.

Figure 1-1 shows a sample command.

```
F DFRMM,CMD=LISTCONTROL ALL
```

Figure 1-1 RMM TSO command from the console

Note: There might be some truncation and line output wrapping when using this function. The effective line length of SYSLOG is less than the 79-column standard on TSO. Whether this results in the loss of significant information depends on the command. Lines that end with “=” are typically truncated, and they might also have information wrapped to the next line.

1.7.2 Improved security control over DFSMSrmm functions

New resource profiles are implemented in z/OS v1R7 DFSMSrmm to better control access to DFSMSrmm resources. It is no longer necessary to have CONTROL access to STGADMIN.EDG.MASTER to run daily DFSMSrmm tasks.

When checking authorization to use RMM subcommands and operands, DFSMSrmm checks in the following sequence:

1. CONTROL access to STGADMIN.EDG.MASTER. If the user is authorized, no further checking is performed.
2. Next, DFSMSrmm checks for specific subcommand operands and for each operand that requires specific authorization checks for the required access. If the resource is not protected, authorization continues with the next step. If the resource is protected, but the user is not authorized, the subcommand fails.
3. Finally, DFSMSrmm continues with ownership checks and RELEASE and FORCE checking, if required.

1.7.3 Use of large format data sets

Large format data sets can have greater than 65,535 tracks per volume. Before z/OS V1R7, most sequential data sets were limited to 65,535 tracks on each volume, although most hardware storage devices supported far more tracks per volume. To support this hardware capability, z/OS V1R7 allows users to create new large format data sets, which are physical sequential data sets with the ability to grow beyond the previous size limit. Large format data sets are not required to have greater than 65,535 tracks at initial allocation, but once allocated in large format, they can expand to greater than 65,535 tracks.

1.7.4 Enterprise enablement

In DFSMS V1.7, the direction of extending DFSMSrmm to the enterprise continues. Extending DFSMSrmm to the enterprise is accomplished through enabling the object-oriented interface to the RMM API to be used via Web services and creating a plug-in adapter for use with Storage Networking Industry Association (SNIA) CIMOM SMI-S.

With z/OS V1R7 and the DFSMSrmm enterprise enablement enhancement, you can use the high-level language API as a Web service. This enables the API to be used from any system or platform that can run Java, C++, or any language that supports the Web services standards. Now, it is as if the high-level language API is available as a locally callable program. A single call to the application programming interface to run a subcommand and receive all the output is all that is needed.

1.8 Changes introduced with z/OS V1R6 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R6.

1.8.1 ISPF dialog enhancements

The DFSMSrmm ISPF dialog has been enhanced for better usability. You can perform these tasks:

- Request DFSMSrmm prime panels with saved variables
- Create a list of DFSMSrmm TSO subcommands using the dialog
- Use new line operators in dialog lists to better manage DFSMSrmm resources
- Specify a new user option to define the type of output station where you want your system-managed cartridges ejected

1.8.2 DFSMSrmm client/server support

DFSMSrmm can be used in a client/server environment. Part of this function includes new operator commands to stop and start the subsystem or IP tasks. This function also provides new REXX variables that you can use to write REXX execs to obtain information about your client/server environment.

1.8.3 Enhanced IBM 3592 support

DFSMSrmm provides support for you to use the IBM Total Storage Enterprise Tape System 3592 WORM tapes. Information has been added about the following topics:

- Added new media types MEDIA6, MEDIA7, and MEDIA8
- Added operand values for ADDVOLUME, CHANGEVOLUME, and SEARCHVOLUME
- Added new return codes and reason codes
- Added new REXX variables and updated several existing variables
- Updated command syntax diagrams for ADDVOLUME, CHANGEVOLUME, and SEARCHVOLUME

1.8.4 Command reference summary moved to DFSMSrmm guide

The *z/OS DFSMSrmm Command Reference Summary* is no longer available as a separate document. The command summary information has been added to the *DFSMSrmm Guide and Reference,* SC26-7404, as an appendix.

1.8.5 C++ API with XML output option

You can use C++ and other high-level programming languages to write programs to obtain information about DFSMSrmm resources. You use the same DFSMSrmm subcommand strings that you can use with the EDGXCI application programming interface (API).

You can receive output as structured field introducers or in Extensible Markup Language (XML). The XML output contains data and tags to define the data. DFSMSrmm provides a schema called rmmxml.xsd that contains the definitions for the XML. For XML output, DFSMSrmm converts the data to characters in Unicode format as defined in the XML schema file for the DFSMSrmm resources.

1.9 Changes introduced with z/OS V1R5 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R5.

1.9.1 Enhanced multilevel security

Enhanced multilevel security to support the RACF name hiding function is now provided:

- DFSMSrmm provides added protection for data sets and volumes by supporting the RACF name hiding function.
- Recognition of RACF SETROPTS MLNAMES support has been added since introducing this feature in z/OS 1.3.

1.10 Changes introduced with z/OS V1R4 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R4.

1.10.1 Back up the DFSMSrmm control data set and journal at any time

You can back up the DFSMSrmm control data set and journal at any time. New JCL examples have been added that show how to request a backup of the DFSMSrmm control data set and how to clear the journal.

Backup has been enhanced to run with other Inventory Management (housekeeping) functions, such as the VRSEL, RPTEXT, or XRPTEXT process.

1.10.2 Checking authorization to ignore volumes

You can control whether DFSMSrmm manages or ignores volumes based on whether they are defined to DFSMSrmm or not. You can use RACF profiles to separate authorization checking for volumes that are defined to DFSMSrmm and that are not defined to DFSMSrmm. To support this new processing, two new resources in the RACF class FACILITY are added:

- STGADMIN.EDG.IGNORE.TAPE.RMM.*VOLSER*
- STGADMIN.EDG.IGNORE.TAPE.NORMM.*VOLSER*

1.10.3 Duplicate volume support

You can use volumes with duplicate volume serial numbers. Information and examples about how to define volumes with duplicate volume serial numbers have been added. You can also use the DFSMSrmm EDGINERS utility to initialize duplicate volumes.

Duplicate volume support allows you to perform these tasks:

- Manage duplicate volume serials without using EDGUX100
- Use any number of duplicate volumes with the same VOL1 label
- Enable support for Tape Open input and output processing
- Return to scratch automated to relabel duplicate VOLSERs to match VOL1

1.10.4 Enhanced multi-level security

DFSMSrmm provides added protection for data sets and volumes by providing command authorization support using the DFSMSrmm EDGRMMxx parmlib OPTION COMMANDAUTH operand.

Multilevel security support allows these functions:

- COMMANDAUTH PARMLIB member to control calls to RACF DATASET class, RACF TAPEVOL class, or both.
- COMMANDAUTH defaults to maintain access by OWNER for previous release compatibility.

1.11 Changes introduced with z/OS V1R3 DFSMSrmm

This section describes the enhancements to DFSMSrmm introduced with z/OS V1R3.

1.11.1 CDS utilization displays now available

CDS utilization is available via the LISTCONTROL command in the control record.

1.11.2 Journal utilization and status displays now available

Journal utilization is now available via the LISTCONTROL command in the control record.

1.11.3 Movement date maintained for ON LOAN volumes

Volumes Movement Tracking Date now maintains date tape that was placed On Loan.

1.11.4 Enhanced EJECT support

Enhanced EJECT support allows for these functions:

- Eject of volume synchronization with the tape configuration database (TCDB)
- Eject failed if control data set (CDS)/TCDB status mismatch
- DFSMSrmm checking of whether a volume is in an EJECT queue

1.11.5 Enhanced foreign tape authorization

Foreign tape authorization has been enhanced to offer the following new functions:

- New RACF FACILITY classes:
  - STGADMIN.EDG.IGNORE.TAPE.**RMM**.*
  - STGADMIN.EDG.IGNORE.TAPE.**NORMM**.*
- Control of how foreign tapes are to be processed, either with volumes defined in DFSMSrmm or volumes out of DFSMSrmm control

1.11.6 3590-H support

3590-H includes support of the following items:

- D/T3590-H support for 384TRACK recording format
- EDGINERS updates to the media type and recording format

1.11.7 Report Generator enhancements

The Report Generator provides these functions:

- Field-to-field compare, including position override
- &TODAY with relative values (+/-); Days, Months, and Years
- Break TOTALS when using Grouping
- JCL skeleton now named in Report Tool Table
- Optional Extract step in generated JCL
- Report Definition Table shows Selection fields

1.11.8 Expiration Date field added to DFSMSrmm Data Set record

The addition of the Expiration Date field to the DFSMSrmm Data Set record allows for the following function:

- Expiration Date and Original Expiration Date now supplied in Data Set record
- Report extract and extended report extract file changed to add Expiration Date and Original Expiration Date
- Enhanced DFSMSrmm TSO command support to allow for entry of a date or a retention period for adding and changing data sets and volumes

1.11.9 Multivolume dialog support

The multivolume dialog support provides these options:

- A dialog option to process the entire multivolume set for Change and Release updates
- Dialog informational messages that are issued for multifile, multivolume, or multifile/multivolume
- Dialog L line command, which has been enhanced to list all data sets on the set in sequence

1.11.10 Special character support

Special characters are now supported for the following DFSMSrmm entry types:

- VOLSER
- RACK
- POOL

These characters are also supported on the REJECT parameter. There is no support for special characters for the BIN entry type. The special characters now supported are: , . ‘ / ( ) * & + - = and blank.

Special considerations

Note the following considerations for special character support:

- For volumes that reside in IBM automated tape libraries, only alphanumeric characters are supported (with bar code labels) unless the unlabeled tape facility is used. When the unlabeled tape facility is used, the alphanumeric characters, plus hyphen (-), number or pound sign (#), ampersand (&), dollar sign ($), and at sign (@) are supported.
- Leading blanks are not allowed. DFSMSrmm does not support a leading blank on the VOLSER.
- Asterisks are allowed anywhere in a VOLSER:

  `AA001*`, `*A0001`, and `A*0001` are all valid VOLSERs.

  Using an asterisk in a VOLSER might lead to unexpected results when using the RMM dialog. The RMM dialog treats the asterisk as a wildcard character and therefore returns the actual VOLSER requested and any others that match the mask specified.
- The ADDVOLUME *VOLSER* COUNT(*n*) supports fewer than six characters in the VOLSER field. The following ADDVOLUME subcommand adds volumes A0001 through A0015:

  ```
  ADDVOLUME A0001 COUNT(15)
  ```

- RACKs can now be fewer than six characters.
- Use the TPRACF(N) option for special character VOLSERs. You must protect them outside RMM with a generic RACF TAPEVOL profile, if required. This is because RACF does not support special characters in tape volume VOLSERs for the TAPEVOL class.

1.11.11 Changed messages for improved diagnostics

There are two changed messages to improve error diagnostics:

- The catalog name is included in CATSYNCH VERIFY messages:

  ```
  EDG2236I DATA SET DQQ.SAMMEL.ESA.TREND.G0001V00 VOLUME B51987 DSSEQ 1 IN
  CATALOG UCAT.X IS NOT DEFINED TO DFSMSrmm
  ```

- Pool information is added to the tape reject message:

  ```
  EDG4021I VOLUME volser REJECTED, IT IS NOT IN AN ACCEPTABLE SCRATCH POOL. rtype
  rvalue REQUESTED. mtype mvalue MOUNTED

  type = SG, PREFIX. value = SG name, pool prefix value
  ```

1.11.12 HELP moved from SYS1.SEDGHLP1 to SYS1.HELP

You no longer need to copy DFSMSrmm TSO/E help into SYS1.HELP because DFSMSrmm TSO/E help is now included in SYS1.HELP with the rest of DFSMS. When all systems in your sysplex are on z/OS V1R3 DFSMS, SYS1.SEDGHLP1 can be removed from your concatenations and deleted from the systems.

2

Introduction to DFSMSrmm

Data Facility System Managed Storage removable media manager (DFSMSrmm) is a full function tape management system that is available as a functional component of z/OS. It enables you to manage your removable media as an enterprise library across systems that can share disk.

DFSMSrmm can manage all of your tape volumes and the data sets on those volumes. It protects tape data sets from being accidentally overwritten, manages the movement of tape volumes between libraries and vaults over the life of the tape data sets, and handles expired and scratch tapes, all according to policies that you define. Plus, DFSMSrmm manages other removable media that you define to it. For example, it can record the shelf location for optical disks and track their vital record status. DFSMSrmm does not yet automatically record information for optical volumes.

This chapter provides a brief introduction to DFSMSrmm basic concepts and functions. By the end of this chapter, you will have a basic understanding of DFSMSrmm.

2.1 Environment

All data set, volume, and policy information is kept in the DFSMSrmm control data set (CDS). The CDS is a Virtual Storage Access Method (VSAM) key-sequenced data set (KSDS) that contains all inventory information, and can be either an extended format (EF), or a non-extended format VSAM data set. The only other dedicated data set is the DFSMSrmm journal. This is a sequential data set that is used to record all updates to the CDS. When not shared with a z/OS release earlier than z/OS V1R12, the DFSMSrmm journal can be placed in an extended addressing space (EAS) on an extended address volume (EAV).To tailor your processing options, create the EDGRMMxx PARMLIB member. Figure 2-1 shows the DFSMSrmm environment.

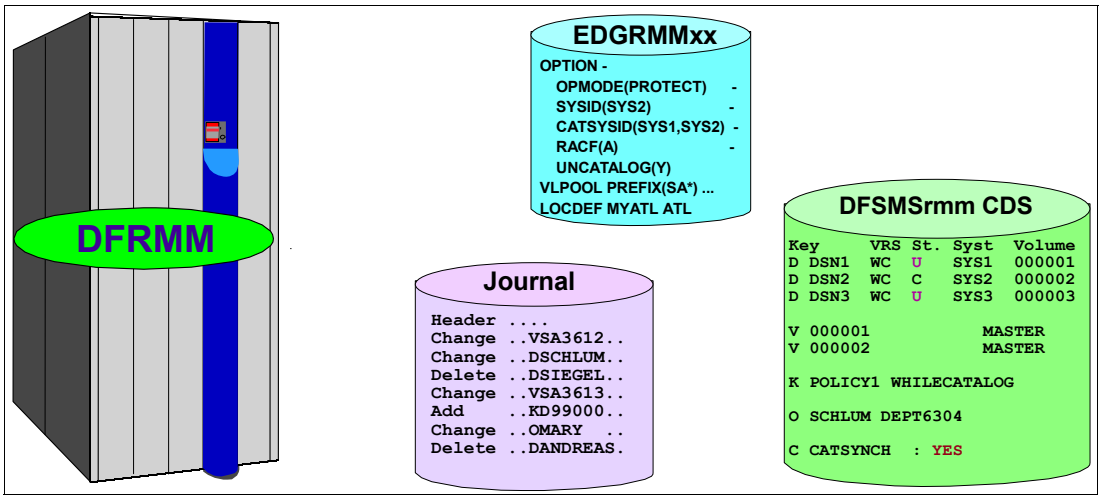


Figure 2-1 DFSMSrmm environment

Note: The extended format (EF) VSAM data sets can offer the following benefits:

- Data striping
- Data compression
- VSAM extended addressability
- Partial space release
- System-managed buffering
- CDS size greater than 4 GB

All data sets used by DFSMSrmm can be eligible for allocation in the extended addressing space of an extended address volume (EAV). This includes the DFSMSrmm journal and dynamically allocated temporary files.

All DFSMSrmm interactions take place under the control of the subsystem. Utilities are provided to start the HOUSEKEEP function, so that you can schedule DFSMSrmm work through your production scheduling systems. One benefit of running as a subsystem is evident in the support for system-managed tape: DFSMSrmm uses a subsystem-to-subsystem interface to provide as much support as possible without involving the operator.

DFSMSrmm supplies ISPF panels as well as TSO commands. Therefore, almost anything you can do online, you can also do in batch through TSO command execution.

DFSMSrmm provides utilities to manage your inventory, create reports, maintain the DFSMSrmm CDS, and erase and initialize volumes.

An application programming interface (API) is provided to enable user-written programs to access the DFSMSrmm subsystem, and allows interrogation and update of the subsystem.

Use the EDGHSKP utility, with the DFSMSrmm subsystem active to run inventory management activities:

- Vital record processing, to determine which data sets to retain and which volume moves are required, based on vital record specification (VRS).
- Expiration processing, to identify volumes ready to be released and returned to scratch.
- Storage location management processing, to set a destination for a volume. Optionally, run storage location management to assign shelf locations in storage locations for volumes that are being sent out or returned to the removable media library.
- Backing up the CDS and the journal, and clearing the journal.
- Creating an extract data set for report generation.

EDGHSKP also has some trial run and predictive features that allow a trial run of inventory management vital record processing without making any changes to the control data set or journal.

Use the EDGUTIL utility to create, update, and verify the CDS. Use the EDGBKUP utility to back up and recover the CDS and journal. Both EDGUTIL and EDGBKUP can be executed independently of the subsystem.

2.1.1 CDS backup and recovery

When backup is performed using the DFSMSdss DUMP command, DFSMSrmm backs up the CDS and then the journal. DFSMSrmm resets the journal data set and discards journal records if the backup completes successfully. To restore the CDS using the latest CDS backup, both the latest journal backup and the current journal must be used for forward recovery.

DFSMSrmm clears the journal when the CDS and journal are backed up. Clearing the journal data set prevents the journal from becoming full, and reduces the risk of losing updates to the CDS. Backing up the CDS and the journal using DFSMSdss allows you to write the data directly to tape. Using DFSMSdss concurrent copy allows simultaneous updates to the DFSMSrmm CDS so that other tape activity need not be interrupted.

Note: With DFSMSrmm release z/OS 1.10 or higher, you can request that DFSMSrmm back up the control data set and the journal, using the DFSMSdss COPY command.

2.2 Reports

You can obtain information and create reports:

- DFSMSrmm ISPF dialog, TSO RMM commands, or the API
- EDGAUD, EDGJRPT JCL, and EDGRPTD report utilities
- EDGRRPTE sample REXX exec to create 17 standard reports
- DFSORT's ICETOOL utility
- DFSMSrmm Report Generator

Searches and lists

You can search online by using either the DFSMSrmm ISPF dialog or TSO RMM subcommands to create lists of resources and display information recorded in the DFSMSrmm CDS. Here are some examples:

- Operators can create lists of scratch volumes to be pulled for use.
- Tape librarians and system programmers can create lists of software products and the volumes on which they reside.
- General users can create lists of volumes they own, as shown in the example in Figure 2-2.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- -------
DV0001 MHLRES1        2003/182   2003/182   SHELF    0     MR 0
DV0002 MHLRES1  DV0002 2003/182   2003/182   SHELF    1     MV
DV0003 MHLRES1  DV0003 2003/182   2003/182   SHELF    1     MV
DV0004 MHLRES1  DV0004 2003/182   2003/182   SHELF    1     MV
DV0005 MHLRES1  DV0005 2003/182   2003/182   SHELF    1     MV
DV0006 MHLRES1  DV0006 2003/182   2003/182   SHELF    1     MV
DV0007 MHLRES1  DV0007 2003/182   2003/182   SHELF    1     MV
DV0008 MHLRES1  DV0008 2003/182   2003/182   SHELF    1     MV
DV0009 MHLRES1  DV0009 2003/182   2003/182   SHELF    1     MV
DV0010 MHLRES1  DV0010 2003/182   2003/182   SHELF    1     MV
SEARCH COMPLETE - MORE ENTRIES MAY EXIST
10         ENTRIES LISTED
```

Figure 2-2 List of volumes owned by a single user

With DFSMSrmm, you can use the TSO RMM SEARCH subcommands with the CLIST operand to create a data set of executable subcommands. For example, you can create subcommands to confirm volume movement for volumes identified during a SEARCHVOLUME request.

Report utilities

You can create several types of reports using DFSMSrmm reporting utilities. Use EDGRPTD, EDGJRPT, or the DFSMSrmm Report Generator to create movement and inventory reports, and EDGAUD to create security and audit reports. EDGRPTD uses the DFSMSrmm extract data set as input. EDGJRPT and the Report Generator use the extended report extract file. EDGAUD uses System Management Facility (SMF) records as input.

You can use the reports to perform these tasks:

- Identify volumes that need to be moved between the removable media library and storage locations
- Determine your volume inventory in the removable media library and storage locations
- Identify volumes that are in transit or need to be marked as moved
- Identify all accesses to volumes and changes to information recorded in the DFSMSrmm CDS
- Separate volumes waiting to return to scratch from those that are private or have other release actions pending

You can use DFSORT or a similar program to generate a formatted report using the information in the EXTRACT data set created by the EDGHSKP utility. Also, the ACTIVITY file and the EDGJACTP sample JCL can be used to create a report from it. For example, you can produce an extract data set listing all volumes to be used on VM with information about volume owners. Then, use DFSORT's ICETOOL utility to sort the information by volume and produce a report, complete with title and header information.

2.2.1 Security

You can choose the authorization levels of users for all DFSMSrmm functions. DFSMSrmm uses the MVS system authorization facility (SAF) for its authorization checking. You define DFSMSrmm resources to z/OS Security Server Resource Access Control Facility (RACF) for use during authorization checking. DFSMSrmm can create volume profiles, change them, and delete them on registration, expiration, or release of volumes. DFSMSrmm optionally supports security profiles in the RACF DATASET and TAPEVOL classes as a way to authorize the use of certain RMM TSO subcommands against data set and volume information defined in the CDS.

You can use the access list that DFSMSrmm provides to set the access list in RACF, as well as for authorization checking on non-RACF systems. To display the DFSMSrmm access list, use the TSO RMM LISTVOLUME subcommand or the DFSMSrmm ISPF dialog volume list function. You can also find the access list in the volume records in the report extract data set.

DFSMSrmm provides automatic security classification through installation-specified criteria based on data set names. It provides the following control of classified volumes:

- Audit trail of access and change of status through SMF.
- This audit trail produces information about the RACF user ID, group, and job name.
- Operator confirmation of use.
- Erasure of data when a volume is released before it is returned to scratch status.

DFSMSrmm provides the following ways of optionally keeping an audit trail for volumes defined to it:

- Control data set information
- SMF audit records
- RACF audit information

DFSMS V1.8 provides new options for securing tape data sets using System Authorization Facility (SAF). These are designed to allow you to define profiles to protect data sets on tape using the DATASET class without the need to activate the TAPEDSN option or the TAPEVOL class. DFSMS also provides options you can use to specify that all data sets on a tape volume need to have common authorization and that users are authorized to overwrite existing files on a tape volume.

Note: For optimum tape security, use the combined capabilities of DFSMSrmm, DFSMSdfp, and RACF.

2.3 How is DFSMSrmm structured

In Figure 2-3 on page 28, you can see the basic DFSMSrmm structure. DFSMSrmm runs in its own subsystem address space inside the z/OS operating system. It communicates directly with the Subsystem Interface (SSI), allowing DFSMSrmm to fully control the tape usage in the system.

A user address space is any task requesting information from DFSMSrmm or any user of tape. For example, this can be a batch job creating a tape data set, or a user listing owned volumes.

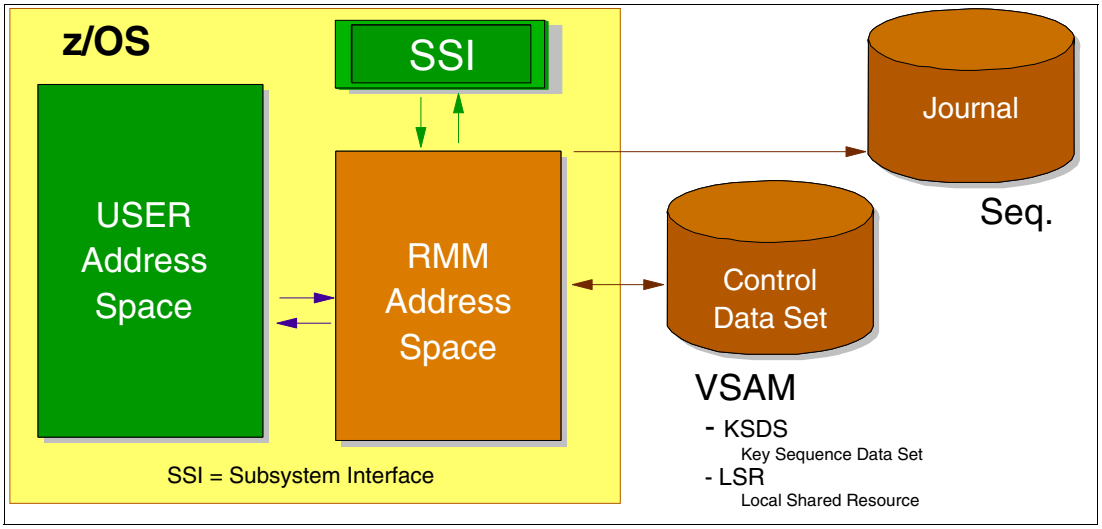


Figure 2-3 DFSMSrmm structure

DFSMSrmm uses the following data sets for its activity:

- Control data set

  The control data set is a VSAM key-sequenced data set that contains the complete inventory of the removable media library and storage locations. DFSMSrmm records all changes made to the inventory, such as adding or deleting volumes in the control data set. The control data set needs to be RACF-protected to prevent unauthorized access to the data. DFSMSrmm requires CONTROL access to the control data set.

- Journal

  The journal is a sequential data set, which contains a record of all changes made to the control data set since the last backup.

For information about these data sets, such as structures, size, and definition, see 3.1, “DFSMSrmm implementation” on page 70.

When multiple systems exist, they can share the same control data set. The control data set cannot be a SYS1 data set if the control data set is to be shared. Figure 2-4 on page 29 shows the structure in this environment.

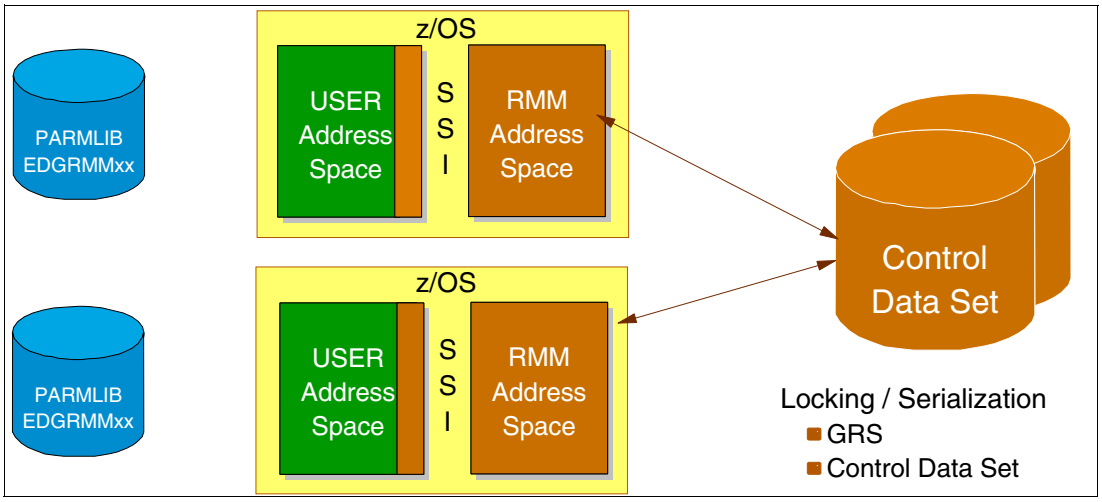


Figure 2-4 DFSMSrmm in a multisystem environment

When multiple systems share one CDS, you can use Global Resource Serialization (GRS) resource definition to solve the possible contention problems. For details about how to customize GRS reserve and release actions for DFSMSrmm, see “GRSRNLxx (optional): GRS resourcename list” on page 101.

2.4 What are the DFSMSrmm interfaces

DFSMSrmm interacts with different system and non-system components to perform the tape management functions. We introduce three different types of interfaces: software interfaces, user interfaces, and programming interfaces.

Software interfaces are those interfaces between DFSMSrmm and some system components or activities, such as allocation, Open/Close/End of Volume (O/C/EOV) events, and between DFSMSrmm and other software components, such as DFSMSdfp, DFSMShsm, DFSMSdss, RACF, catalog, and Object Access Method (OAM).

User interfaces are interfaces that allow the communication between DFSMSrmm and the user. These user interfaces are the Interactive System Productivity Facility (ISPF) dialogs and TSO commands.

Finally, the programming interfaces allow communication between DFSMSrmm and user applications, or programs, to obtain information from DFSMSrmm or to provide information to DFSMSrmm.

2.4.1 Software interfaces

Figure 2-5 on page 30 shows a general view of what we consider the main DFSMSrmm software interfaces. Using these interfaces, DFSMSrmm either provides support to other users or requests services from them. For detailed information about this topic, see *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

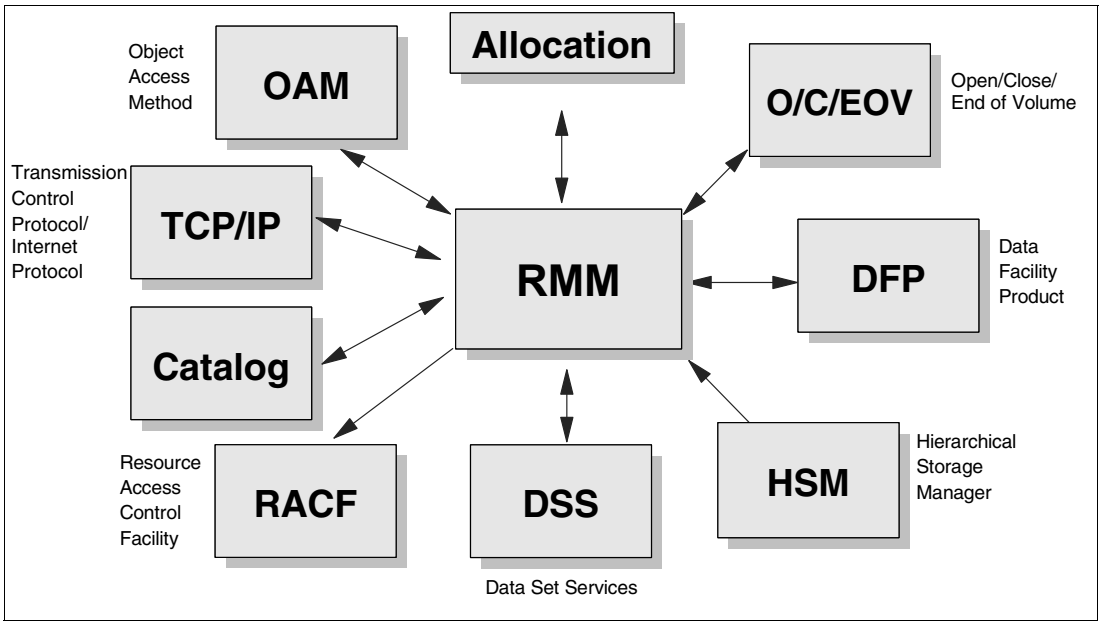


Figure 2-5 Interfaces between DFSMSrmm and other software components

The kinds of information that are exchanged between DFSMSrmm and each of the software interfaces, as well as the activities performed, are listed:

- Allocation

  At allocation time, DFSMSrmm intercepts the mount message, assigns a scratch pool, and finds the rack where the volume resides. After that, it modifies the mount message to finish the response to this request.

- Open/Close/End of Volume (O/C/EOV) interface

  DFSMSdfp provides tape exits for vendor or client use. DFSMSrmm has the capabilities to work with these exits to manage the open, close, and end of volume event of tape. When such events happen, DFSMSrmm performs these actions:

  – Validates the mounted tapes
  – Stores information in the CDS for volume and data sets, if authorized
  – Returns block-ID for high-speed locate
  – Rejects the volume in these situations:
    - A wrong volume is mounted for a specific volume request.
    - A private volume is mounted in response to a scratch request.
    - An attempt is made to read a scratch volume.
    - An attempt is made to overwrite a data set and the data set name does not match.
    - An attempt is made to use a specific scratch volume.
    - The first file of the volume does not match what DFSMSrmm has recorded.
    - An attempt is made to read or write a volume using nonstandard labels.
    - Unauthorized functions are used, such as bypass label processing (BLP), write label, and so on.

- DFSMSdfp interface

  DFSMSrmm provides the EDGMSGEX exit as an interface to DFSMSdfp. DFSMSrmm uses the IGXMSGEX exit, the MSGDISP installation exit of DFSMSdfp, to call EDGMSGEX to retrieve DFSMSrmm information to update tape drive displays and control the use of cartridge loaders.

For system-managed tapes, when the volume is open or closed or an end-of-volume event occurs, the storage management subsystem (SMS) automatic class selection (ACS) routines determine the volume information.

For non-system-managed tapes, DFSMSrmm calls SMS ACS routines to get information about the storage group for volumes and the management class for data sets.

Figure 2-6 shows the differences between these allocations.

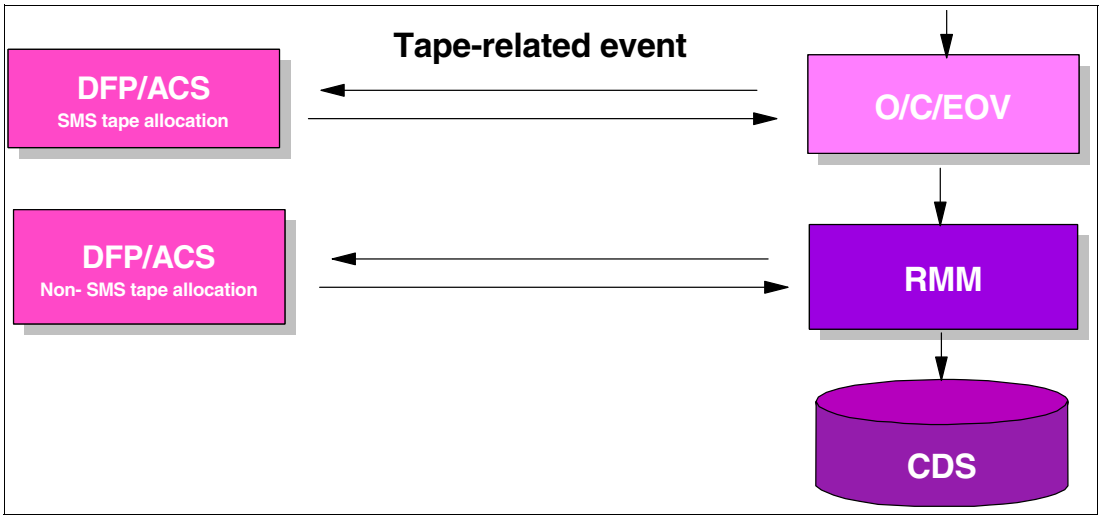


Figure 2-6 SMS tape allocation

- DFSMShsm and Tivoli Storage Manager interface

  DFSMSrmm provides EDGTVEXT and EDGDFHSM programming interfaces for use by products, such as DFSMShsm and Tivoli Storage Manager, when using DFSMSrmm tape management.

  DFSMSrmm treats DFSMShsm like any other user, and retains volumes based on the retention period and VRS. DFSMSrmm recognizes when DFSMShsm is opening tape data sets, and for input processing, tolerates the data set names that DFSMShsm uses as long as the last 17 characters of the data set name match.

  For Tivoli Storage Manager, you need to define policies (VRSs) to cover the Tivoli Storage Manager tape data sets. A single policy specifying the Tivoli Storage Manager high-level qualifier is sufficient to include all Tivoli Storage Manager tape data sets.

- DFSMSdss interface

  DFSMSrmm uses the DFSMSdss user interaction module (UIM) interface to enable the use of DFSMSdss to back up the CDS. When you use DFSMSdss concurrent copy for backup, you minimize the time when DFSMSrmm must prevent updates to the CDS.

- RACF interface

  DFSMSrmm does not provide any security functions itself but relies on installed security products, such as RACF or an equivalent, to process requests. DFSMSrmm uses the MVS Security Access Facility (SAF) interface to perform authorization checks and other security processing.

  You can use RACF to protect DFSMSrmm resources and to authorize DFSMSrmm functions.

  DFSMSrmm also provides support for using RACF standard tape volume security protection. You can use any combination of RACF TAPEVOL and TAPEDSN options to protect tape volumes as well as tape data sets.

- Catalog interface

  DFSMSrmm uses the catalog search interface (CSI) to synchronize the control data set with the catalog for tape data sets. It can also exploit catalog exits to mirror catalog updates in the CDS and uncatalog tape data sets according to the definition in PARMLIB.

- Basic Tape Library Support (BTLS) interface

  If your installation uses BTLS to drive IBM 3494 or IBM 3495 Automated Tape Library (ATL) Dataserver or IBM TS7700 Virtualization Engine operations, you need to notify BTLS when a volume has to return to SCRATCH status. In your tape management system environment, this is done with a batch job that gets information from your tape management system database and then updates the BTLS catalog. In DFSMSrmm, you can use a simple REXX CLIST that synchronizes the DFSMSrmm database with the BTLS catalog.

- SMS interface

  If you have an IBM 3494 or an IBM 3495 Automated Tape Library Dataserver or an IBM TS7700 Virtualization Engine, you must ensure that it works correctly in the DFSMSrmm environment. You have to change the OAM exits, using the DFSMSrmm version, and provide the correct input to the extract program for those volumes that belong to the tape library.

  DFSMSrmm provides programming interface EDGLCSUX to use the OAM. The OAM installation exits used by DFSMSrmm are CBRUXCUA, CBRUXEJC, CBRUXENT, and CBRUXVNL.

  When one of the following events happens, OAM address space uses the corresponding installation exit to send requests to DFSMSrmm:

  - Change volume use attribute
  - Cartridge eject
  - Cartridge entry
  - Volume-not-in-library

  When DFSMSrmm queries information about library or volumes and performs volume-related actions, such as eject, delete volume, or change volume status, DFSMSrmm uses OAM executable macros, such as CBRXLCS, to send requests to the OAM address space.

  - Short-on-scratch processing interface

    DFSMSrmm offers an integrated SMS interface that updates the TCDB automatically.

    In addition, without manual intervention, you can free volumes that are in PENDING RELEASE status when a short-on-scratch condition is detected inside the ATL. Use the PARMLIB option, SCRATCHPROC(*RMMSCR*), where *RMMSCR* is the name of the procedure DFSMSrmm starts to replenish scratch volumes in the ATL.

- High-speed locate function

  DFSMSrmm now records the starting and ending tape block IDs for each data set in the control data set when it is created. With this new technique, DFSMSrmm enables the use of tape block IDs for applications that do not support them (see Figure 2-7 on page 33).

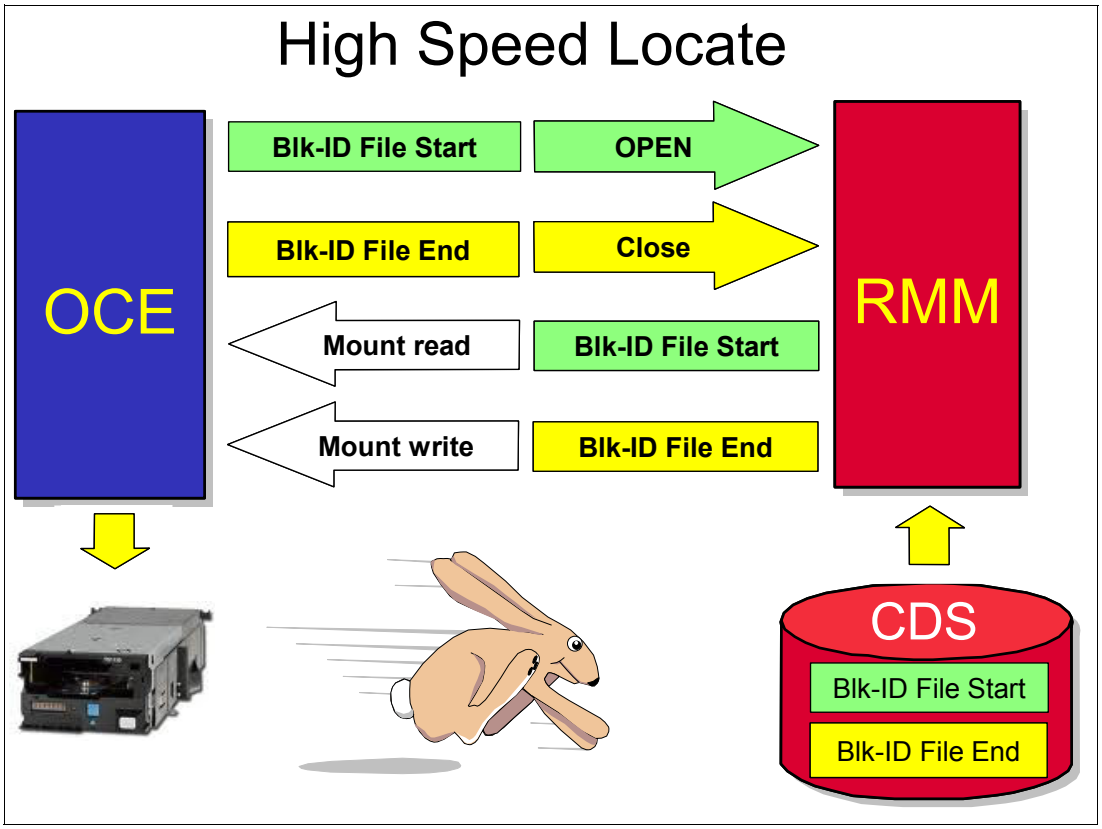


Figure 2-7 High-speed locate overview

The benefit of using this function is that it enables high-speed positioning when a large number of data sets exist in a high capacity tape cartridge.

To gain benefit from this function, you need OS/390 Version 2 Release 10 or higher and DFSMSrmm. The system and DFSMSrmm automatically record and retrieve block IDs, and then use these for high-speed data set positioning whenever it is possible.

2.4.2 User interfaces

DFSMSrmm user interfaces allow the user to interact with DFSMSrmm to perform any of the actions related to tape management. The user can interact with DFSMSrmm in two ways: ISPF panels and TSO commands.

- ISPF dialogs

  The DFSMSrmm ISPF dialog is provided in the ISPF environment. You can tailor it to meet your requirements, such as making DFSMSrmm available on any selection panel you want and controlling the access for different types of users. From the ISPF panels, the user can perform the most commonly used DFSMSrmm functions, such as define, delete, and display volumes, policies, and data sets. Figure 2-8 on page 34 shows the result of a SEARCHVOLUME request to list all volumes beginning with VT* that have a status of MASTER and are stored in the location VTFM001.

```
                    DFSMSrmm Volumes (Page 1 of 2)          Row 1 to 21 of 300
Command ===>                                                 Scroll ===> CSR

Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
   Volume          Assigned   Expir./    S                  Dest-    Tr- Data
S  serial Owner    date       Retn. date R Status  Location ination  ans sets
-- ------ -------- ---------- ---------- - ------- -------- -------- --- -----
   VT0000          2011/324               SCRATCH VTFM001           N   1
   VT0001          2011/324               SCRATCH VTFM001           N   1
   VT0002 STC      2010/293   PERMANENT   VRS     VTFM001           N   1
   VT0003          2011/324               SCRATCH VTFM001           N   1
   VT0004          2011/324               SCRATCH VTFM001           N   1
   VT0005 STC      2010/313   PERMANENT   VRS     VTFM001           N   1
   VT0006 STC      2010/313   PERMANENT   VRS     VTFM001           N   1
   VT0007 STC      2010/313   PERMANENT   VRS     VTFM001           N   1
   VT0008 MHLRES7  2011/324   2011/333    MASTER  VTFM001           N   1
   VT0009 STC      2010/313   PERMANENT   VRS     VTFM001           N   1
   VT0010 STC      2010/286   PERMANENT   VRS     VTFM001           N   1
   VT0011 STC      2010/286   PERMANENT   VRS     VTFM001           N   1
   VT0012 STC      2010/288   PERMANENT   VRS     VTFM001           N   1
   VT0013 MHLRES7  2011/324   2011/333    MASTER  VTFM001           N   1
   VT0014 STC      2010/288   PERMANENT   VRS     VTFM001           N   1
   VT0015 MHLRES7  2011/324   PERMANENT   VRS     VTFM001           N   1
   VT0016 MHLRES7  2011/324   PERMANENT   VRS     VTFM001           N   1
   VT0017 MHLRES7  2011/325   2011/334    MASTER  VTFM001           N   1
   VT0018 STC      2010/293   PERMANENT   VRS     VTFM001           N   1
   VT0019 STC      2010/293   PERMANENT   VRS     VTFM001           N   1
   VT0020 STC      2010/293   PERMANENT   VRS     VTFM001           N   1
```

Figure 2-8 ISPF SEARCHVOLUME result

- TSO commands

  DFSMSrmm allows you to use a TSO command to define resources, manage resources, and create reports. You use the command shown in Figure 2-9 to add a new VRS.

```
RMM ADDVRS DSN('YCJRES4.TEST1.**') DAYS COUNT(1)
```

Figure 2-9 RMM ADDVRS subcommand

- TSO commands from the console

  The MVS modify command can now be used to issue an RMM TSO command from the console. The command output is returned to the console and system log.

  Figure 2-10 shows a sample command.

```
F DFRMM,CMD=LISTCONTROL ALL
```

Figure 2-10 RMM TSO command from the console

2.4.3 Application programming interface

DFSMSrmm provides an API that allows clients to write programs to obtain services from DFSMSrmm. Figure 2-11 shows the relationship between the DFSMSrmm API and other software components. Using the API, you can issue any of the RMM TSO subcommands from an assembler program.

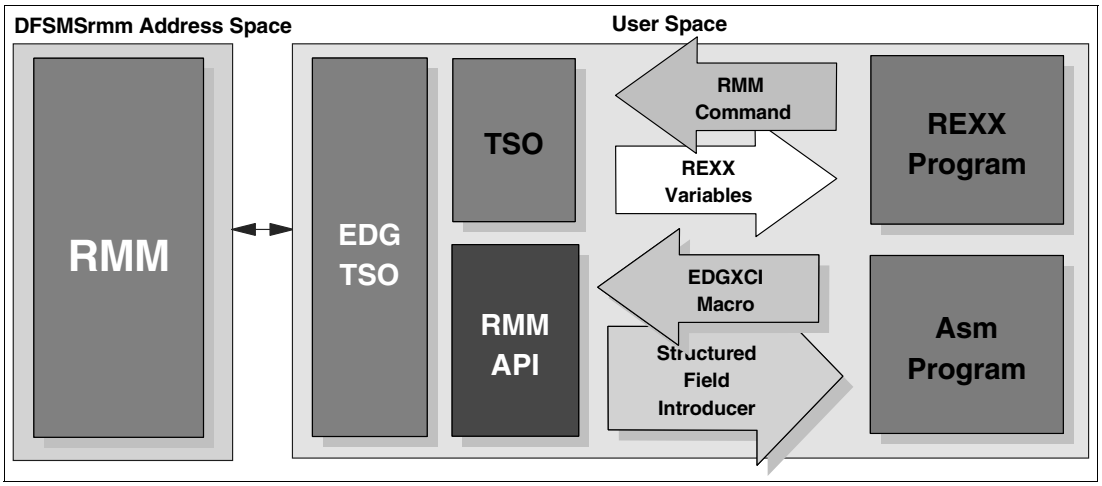


Figure 2-11 DFSMSrmm API

You can use the API to obtain information about DFSMSrmm resources and use the data to create reports or to implement automation. To use the DFSMSrmm API, you must have High Level Assembler installed on your system, and code assembler language programs. For more information about the API, see *DFSMSrmm Application Programming Interface*, SC26-7403.

With DFSMSrmm, you can use C++ and other high-level programming languages to write programs to obtain information about DFSMSrmm resources. You use the same DFSMSrmm subcommand strings that you can use with the EDGXCI application programming interface.

2.4.4 TCP/IP interface

DFSMSrmm uses the TCP/IP interface when implementing the RMMplex (for more information, see 2.13, “Implementing the RMMplex” on page 66). This enables DFSMSrmm to share a control data set between multiple systems without having a shared DASD environment.

IPV6 supports networking strategy

DFSMSrmm on z/OS V1R12 and later releases is an IPv6-enabled application that supports both IPv4 and IPv6 sockets. You can continue to use IPv4 on all systems or you can run a mixed environment with one or more V1R12 systems using IPv6 and other systems using IPv4. After all systems are V1R12, you have the option of moving all systems to IPv6. In a mixed environment, dual-mode IP stacks are required. You can use IP addresses that are compliant with either IPV4 or IPV6.

Note: DFSMSrmm client/server processing is dependent on Internet Protocol V4.

Figure 2-12 on page 36 shows an example of an IPV6 address.

```
1080:0:0:0:8:800:200C:417A
```

Figure 2-12 IPV6 address

2.5 Library and storage location management

You decide where to store your removable media according to how often the media is accessed and for what purpose it is retained. For example, you might keep volumes that are frequently accessed in an ATL, and you probably use at least one *storage location* or vault to retain volumes for disaster recovery and audit purposes. You might also have locations where volumes are sent for further processing. These locations might be data centers within your company or customer and vendor sites.

DFSMSrmm manages the following components and locations:

- Removable media library, which contains all tape and optical volumes that are available for immediate use and includes the shelves where they reside. A removable media library usually includes other libraries:
  - System-managed tape libraries; for example, the IBM 3494 and IBM Virtual Tape Server (VTS) Automated Tape Library Dataserver models, the IBM TS7700 Virtualization Engine, and manual tape libraries.
  - Non-system-managed tape libraries, IBM System Storage® VTF® Mainframe, vendor robotic systems, or traditional tape libraries.
- Storage locations that are not part of the removable media library because their volumes are not generally available for immediate use. Storage locations are typically used to store removable media that are kept for disaster recovery or vital records.
- Loan locations, which are used to track volumes sent to third parties.

Figure 2-13 shows the relationship of the libraries and storage locations.

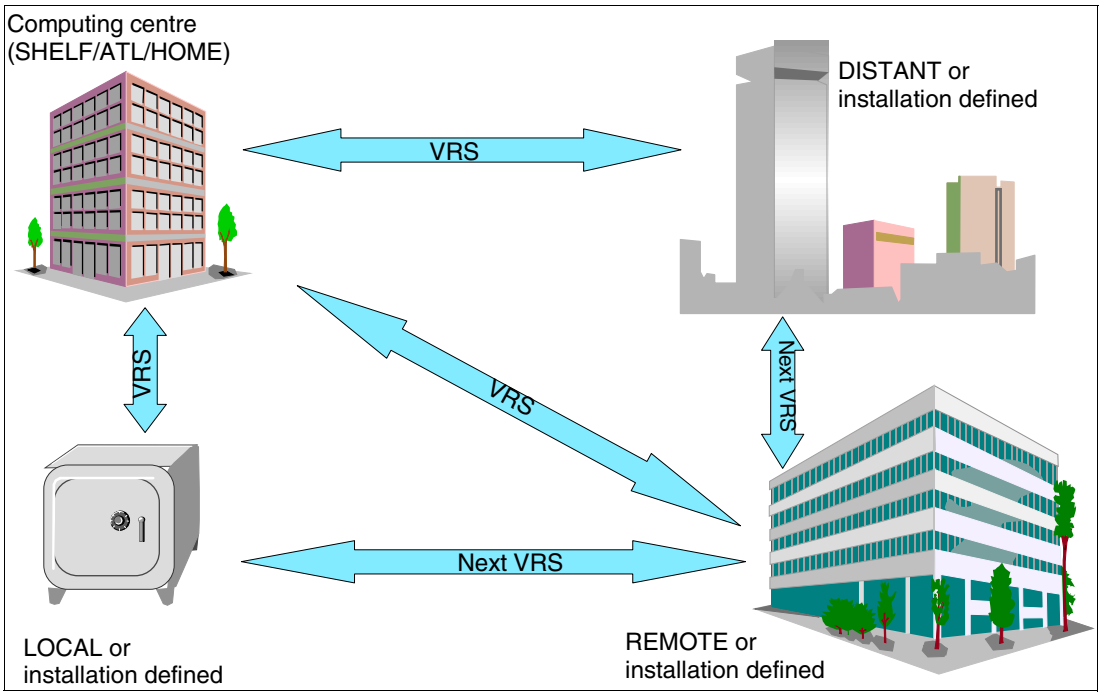


Figure 2-13 Removable media libraries and storage locations

2.5.1 Removable media library

A *removable media library* contains all the tape and optical volumes that are available for immediate use, including the shelves where they reside.

A removable media library usually includes other libraries: *system-managed libraries*, such as *automated* or *manual tape libraries*; and *non-system-managed libraries*, containing the volumes, shelves, and drives that are not in an automated or a manual tape library.

In the removable media library, you can store your volumes in shelves, where each volume occupies a single *shelf location*. This shelf location is referred to as a *rack number* in the TSO RMM subcommands and in the DFSMSrmm ISPF dialog.

A rack number matches the volume's external label. DFSMSrmm uses the volume serial number to assign a rack number when adding a volume, unless you specify otherwise. The format of the volume serial you define to DFSMSrmm must be one to six alphanumeric characters. The rack number must be six alphanumeric or national characters. This information is illustrated in Figure 2-14 on page 38.

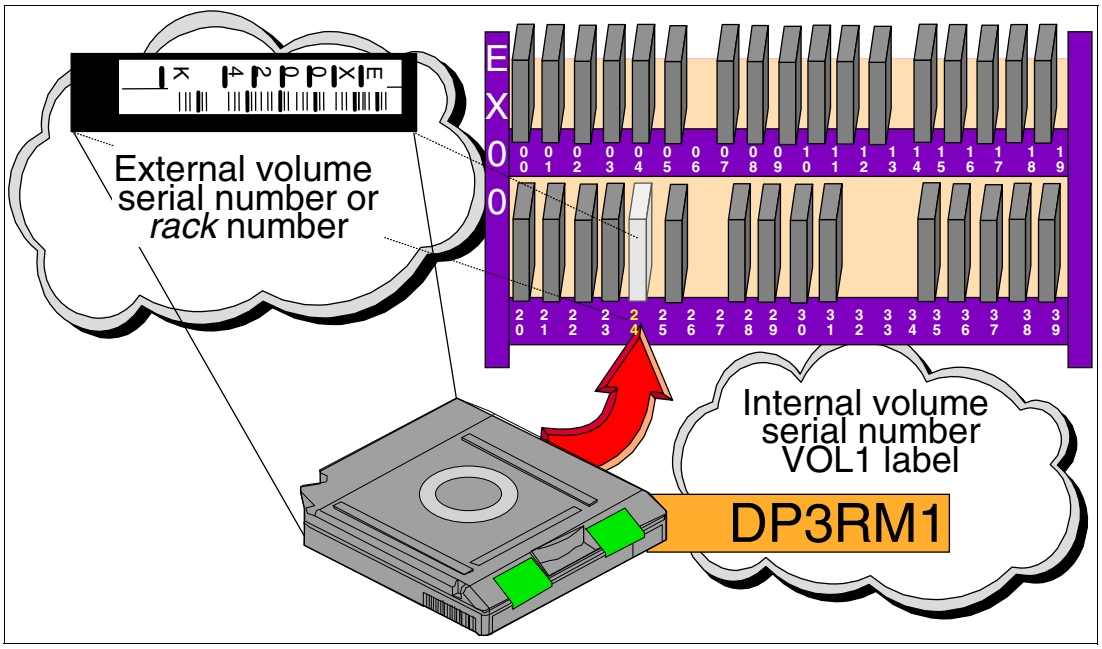


Figure 2-14 Internal labels, external labels, and rack numbers

System-managed tape library

A system-managed tape library is a collection of tape volumes and tape devices, defined in the tape configuration database (TCDB). The TCDB is an integrated catalog facility (ICF) catalog of type VOLCAT. It is a new SMS control data set containing tape library and tape volume records.

A system-managed tape library can be either automated or manual. You can have several automated or manual tape libraries. Use an installation-defined library name to define each ATL or MTL to the system. DFSMSrmm treats each system-managed tape library as a separate location or destination.

Automated tape library

An *automated tape library* consists of robotics components, cartridge storage areas (or shelves), tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. DFSMSrmm provides support for the automated IBM 3494 Tape Library Dataserver, the IBM 3495 Tape Library Dataserver, and the IBM TS7700 Virtualization Engine.

The IBM 3494 Tape Library Dataserver supports 3490E, 3590, and 3592 tape subsystems. The IBM 3495 Tape Library Dataserver supports 3490, 3490E, and 3590 tape subsystems. The IBM 3494 Tape Library Dataserver can also include the 3494 VTS. DFSMSrmm supports the IBM TS7700 grid and the IBM Peer-to-Peer VTS by ensuring that you cannot use the names of distributed VTS libraries with DFSMSrmm. You must only use the names of consolidated (composite) libraries with DFSMSrmm.

Figure 2-15 on page 39 shows an overview of the system-managed tape environment.

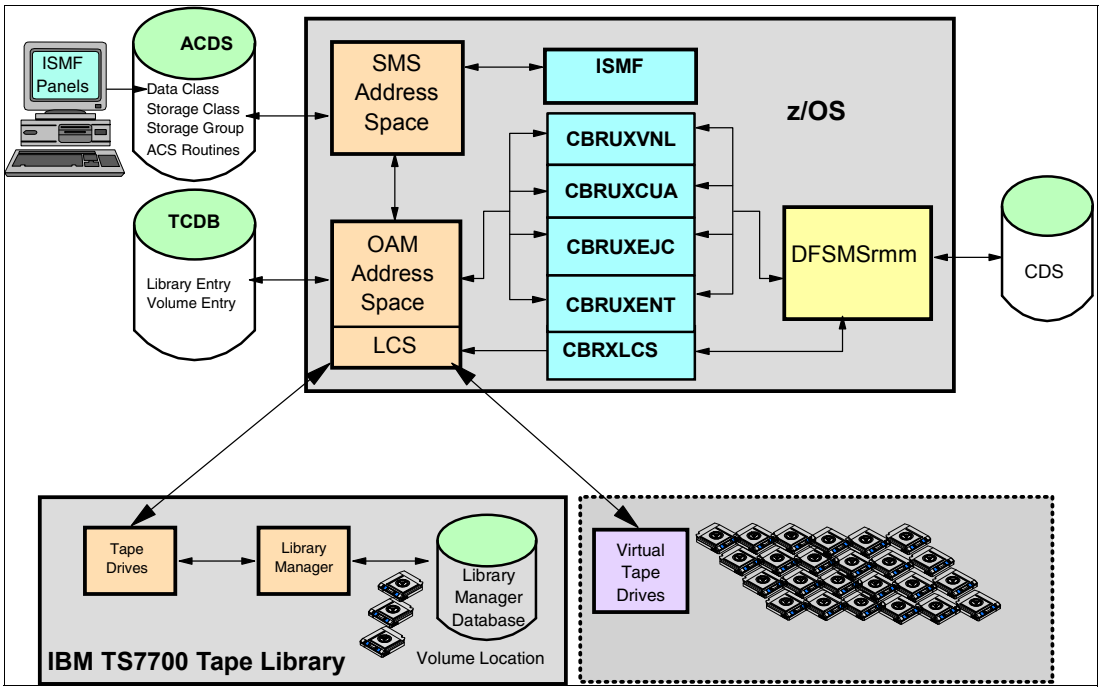


Figure 2-15 System-managed tape overview

Multi-cluster grid terms

This section covers terms that are specific to a multi-cluster grid configuration:

Composite library The composite library is the logical image of the grid that is presented to the host. It is distinctly different from the IBM Virtual Tape Server. A stand-alone cluster TS7700 Virtualization Engine has a five-character hexadecimal Composite Library ID defined. A composite library is presented to the host with both TS7700 Virtualization Engine stand-alone and multi-cluster grid configurations. In the case of a stand-alone TS7700 Virtualization Engine, the host sees a logical tape library with sixteen 3490E tape control units. These units have sixteen IBM 3490E tape drives each, and they are attached through two or four IBM FICON® channel attachments.

Figure 2-16 on page 40 illustrates the host view of a three-cluster grid configuration.

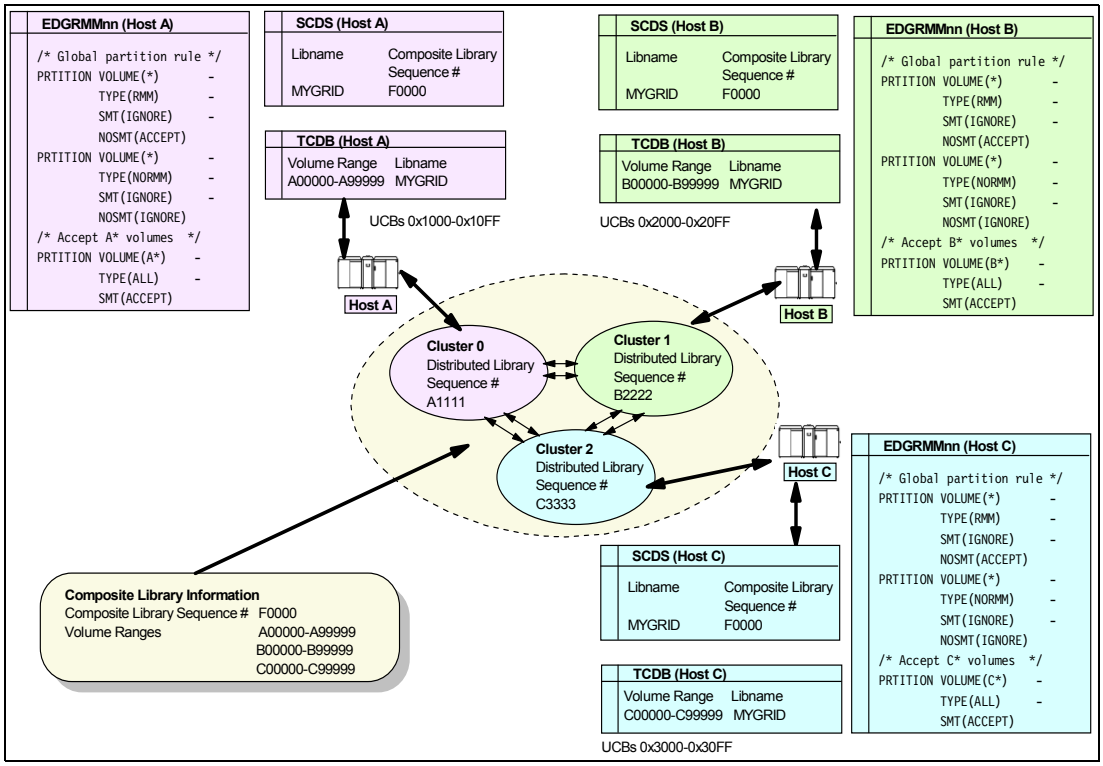


Figure 2-16 TS7700 Virtualization Engine three-cluster grid configuration

Important: A Composite Library ID must be defined both for a multi-cluster grid and a stand-alone cluster. For a stand-alone cluster, the Composite Library ID must not be the same as the Distributed Library ID. For a multiple grid configuration, the Composite Library ID must differ from any of the unique Distributed Library IDs. Both the Composite Library ID and Distributed Library ID are five-digit hexadecimal strings.

Distributed library

Each cluster in a grid is a distributed library, which consists of a TS7700 Virtualization Engine. In the case of a TS7740 Virtualization Engine, it is also attached to a physical TS3500 tape library.

A distributed library consists of the following cluster hardware components:

- A Virtualization Engine
- A TS7700 tape volume cache
- A 3952-F05 frame
- Attachment to a physical library (TS7740 Virtualization Engine only)
- A number of physical tape drives (TS7740 Virtualization Engine only)

A grid configuration with multiple TS7700 Virtualization Engines must have a distributed library defined at the host for each cluster. Each of the engines must have the Grid Enablement feature installed. The host has sufficient knowledge about the distributed libraries to allow appropriate console message handling from the corresponding cluster. At the host, the distributed library is only defined to SMS.

It is defined using the existing ISMF windows and has no tape devices defined. The virtual tape devices are defined for the composite library only.

Copy consistency point

In a grid environment, you can specify where and when you want to have a copy of your data. Currently, the three settings consist of two copy consistency points and an option to have no copy at all:

RUN The copy occurs as part of the rewind/unload (RUN) operation and completes before the rewind/unload operation at the host finishes. This mode is comparable to the immediate copy mode of the Peer-to-Peer VTS.

Deferred The copy occurs after the rewind/unload operation at the host. This mode is comparable to the deferred copy mode of the Peer-to-Peer VTS.

No Copy No copy is made.

On each cluster in a multi-cluster grid, a copy consistency point setting is specified for the local cluster and one for each of the other clusters. The settings can be different on each cluster in the grid. When a volume is mounted on a virtual tape device, the copy consistency point policy of the cluster to which the virtual device belongs is honored.

Dynamic Grid Load Balancing

Dynamic Grid Load Balancing is an algorithm that is used within the TS7700 Virtualization Engine. It continually monitors and records the rate at which data is processed by each network link. Whenever a new task starts, the algorithm uses the stored information to identify the link that will most quickly complete the data transfer. The algorithm also identifies degraded link performance and sends a warning message to the host.

Manual tape library

A *manual tape library* (MTL) is an installation-defined set of tape drives and a user-defined set of tape volumes with mount capability on those tape drives. The volumes can be physically stored in shelf storage located near the MTL, but since these volumes are specifically defined as residing in the MTL, they are known as *library-resident volumes*. When the volumes are logically ejected from the MTL, they become shelf-resident volumes. In an MTL environment, the operator or tape librarian responds to commands at the MVS console, manually loading and unloading the tape cartridges.

Before a tape cartridge can be used, the tape cartridge must first be logically entered into an MTL. Cartridges can be entered into an MTL through invocation of the CBRXLCS manual cartridge entry (MCE) general-use programming interface, or through invocation of the LIBRARY ENTER command.

DFSMSrmm can initiate the MCE process. DFSMSrmm uses MCE processing to add information about a volume residing in an MTL into the TCDB. The RMM ADDVOLUME or CHANGEVOLUME subcommands with the MEDIATYPE operand can be used to define the volume to the MTL. You must ensure that DFSMSrmm has the correct media type for the volume to avoid allocation and mount problems when the volumes are used (see Figure 2-19 on page 44 and Table 2-1 on page 45).

Note: You can set up an IBM System Storage VTF Mainframe as an MTL.

Non-system-managed tape library

A non-system-managed tape library encompasses all the volumes, shelves, and drives not in an automated or manual tape library. You might know this library as the *traditional* tape library. DFSMSrmm provides complete tape management functions for the volumes and shelves in this traditional tape library. DFSMSrmm defines volumes in a non-system-managed library as being *shelf* resident.

All tape media and drives that the operating system supports are supported in a non-system-managed tape library environment. Using DFSMSrmm, you can fully manage all types of tapes in a non-system-managed tape library, including 3420 reels and 3480, 3490, 3590, and 3592 (TS1120, TS1130, and TS1140) cartridge system tapes.

2.5.2 Storage location

A *storage location* comprises shelf locations that you define to DFSMSrmm. A shelf location in a storage location is identified by a *bin* number. DFSMSrmm manages two types of storage locations: installation-defined storage locations and DFSMSrmm built-in storage locations. For more information about built-in storage locations, see Chapter 6, “Managing Storage Locations”, in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

You can define an unlimited number of installation-defined storage locations. You can use up to eight-character alphanumeric names for a storage location and define the type or shape of the media in the location, and the bin numbers that DFSMSrmm assigns to the shelf locations in the location. You can request DFSMSrmm shelf management when you want DFSMSrmm to assign a specific shelf location to a volume in the location.

You can now define that a storage location is a home location, allowing volumes to be scratched while in that location. You can use a storage location defined as a home location for all volumes stored in a non-IBM robot tape library.

You can also use the DFSMSrmm built-in storage locations, LOCAL, DISTANT, and REMOTE. Although the names of these locations imply their purpose, they do not mandate their actual location. All volumes can be in the same or separate physical locations.

For example, an installation can have the LOCAL storage location onsite, as a vault in the computer room, the DISTANT storage location can be a vault in an adjacent building, and the REMOTE storage location can be a secure facility across town. An installation-defined storage location of SHELF can be onsite, and used for volume movement within a library to make better use of the library. Because DFSMSrmm provides shelf management for storage locations, storage locations can be managed at the shelf location level. Optionally, you can define storage locations that are not shelf-managed.

Although DFSMSrmm automatically shelf-manages built-in storage locations, you must first define the bins you want to use to DFSMSrmm. For bin numbers in built-in storage locations, the numbers are fixed in range, starting at bin number 000001. For installation-defined storage locations, a bin number is any six alphanumeric or national characters in any combination.

2.6 HOME location

With DFSMSrmm, you can define any location as a home location for your volumes. To do this, specify each location in your EDGRMMxx PARMLIB member as shown in Figure 2-17 on page 43.

```
LOCDEF LOCATION(EAST) TYPE(STORAGE,HOME) AUTOMOVE(YES)
```

Figure 2-17 Define a storage location as a home location

When you identify a storage location as a home location, you can manage volumes in a storage location, such as volumes that reside in a LIBRARY location. You can perform these tasks:

- Schedule release actions for volumes when they return to their storage home location.
- Return volumes to scratch status while they reside in their storage home location.
- Allow automated movement when the required location does not match the current location of the volume.

Figure 2-18 shows the choices made between z/OS V1.3 DFSMSrmm and previous DFSMSrmm releases. In previous DFSMSrmm releases, the home location was always called SHELF if it was not an automated or manual tape library.

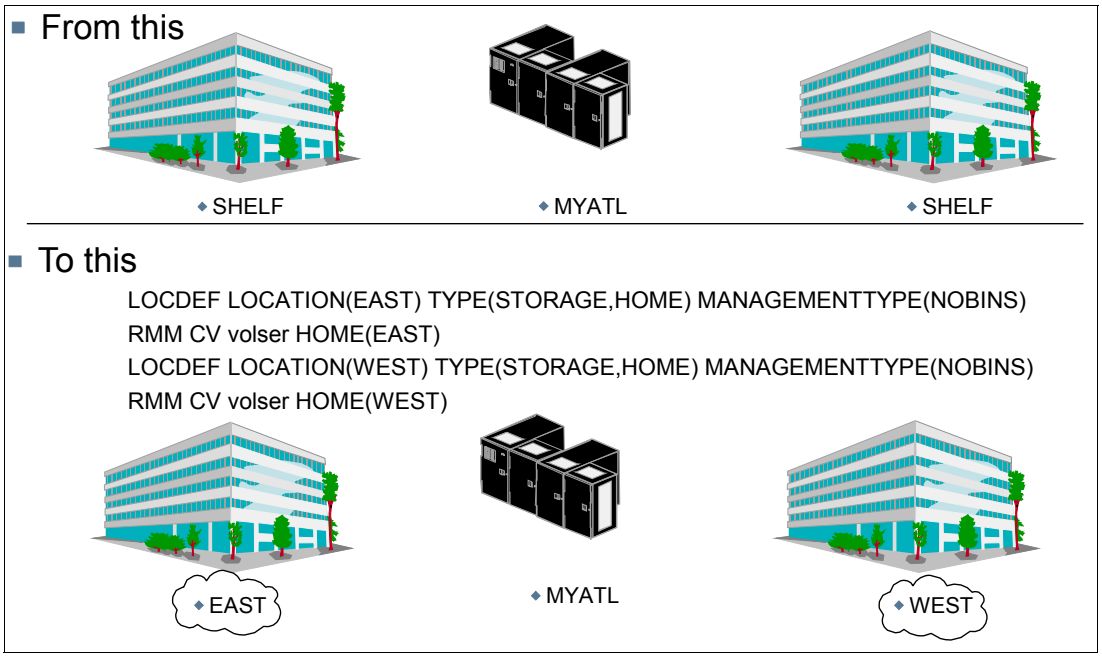


Figure 2-18 Support of different home locations

2.7 Media types

The *media type* specifies the type of physical media of the volumes defined to DFSMSrmm. DFSMSrmm uses the volume's media type as defined to DFSMSrmm when requesting manual cartridge entry; you must ensure that DFSMSrmm has the correct media type to avoid allocation and mount problems when the volumes are later used. You use the RMM ADDVOLUME or CHANGEVOLUME subcommands with the MEDIATYPE operand to set the media type.

Figure 2-19 on page 44 shows the different media types and the terms used in SMS and DFSMSrmm.

Cartridge Type	CST 3480/3490	CST 3490	ECCST 3490	HPCT 3590	HPCT 3590	HPCT 3590	EHPCT 3590	EHPCT 3590	EHPCT 3590
Tracks	18	36	36	128	25	384	128	256	384
Native Compacted	200 MB 600 MB	400 MB 1.2 GB	800 MB 2.4 GB	10 GB 30 GB	20 GB 60 GB	30 GB 90 GB	20 GB 60 GB	40 GB 120 GB	60 GB 180 GB
ATL Media Label	1 or blank	1 or blank	E	J	J	J	K	K	K
OAM Media Type	01	01	02	03	03	03	04	04	04
SMS Tape Category	MEDIA1	MEDIA1	MEDIA2	MEDIA3	MEDIA3	MEDIA3	MEDIA4	MEDIA4	MEDIA4

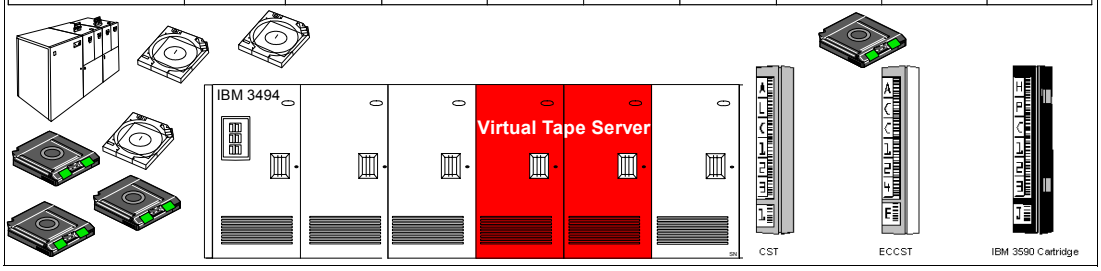


Figure 2-19 Different cartridge media types

Figure 2-20 shows how you can use the different cartridges on all the available tape drives.

Cartridge Type	CST	CST	ECCST	HPCT	HPCT	HPCT	EHPCT	EHPCT	EHPCT
Tracks	**18**	**36**	**36**	**128**	**256**	**384**	**128**	**256**	**384**
3480 without IDRC	read & write	**rewrite***	-	-	-	-	-	-	-
3480/3490 with IDRC	read & write	**rewrite***	-	-	-	-	-	-	-
3490E writes 36 tracks	read **or rewrite***	read & write	read & write	-	-	-	-	-	-
3590 B1x writes 128 tracks	-	-	-	read & write	**rewrite***	**rewrite***	read & write	**rewrite***	**rewrite***
3590 E1x writes 256 tracks	-	-	-	read **or rewrite***	read & write	**rewrite***	read **or rewrite***	read & write	**rewrite***
3590 H1x writes 384 tracks	-	-	-	read **or rewrite***	read **or rewrite***	read **&** write	read **or rewrite***	read **or rewrite***	read & write

Figure 2-20 IBM 3480 and IBM 3490 tape drive matrix

Note: To reformat 384-track format cartridges to 128-track/256-track format, you need the following engineering change (EC) levels when rewriting a tape in a format different from the current format:

- 3590B = D19328
- 3590E = F23079

2.7.1 Tape Cartridges used in the 3592 Tape Drives

The capacity of the 3592 tape cartridge depends on the format used when writing from beginning of tape (BOT). Each tape drive model has different formatting capabilities. The J1A can only read or write in Enterprise Format 1 (EFMT1). The E05 can only read or write in EFMT1 or EFMT2. The E06/EU6 can read tapes formatted in EFMT1, EFMT2, or EFMT3. However, the E06/EU6 can only write using EFMT2 or EFMT3. The E07 can read tapes formatted in EFMT2, EFMT3, or EFMT4. However, the E07 can only write in EFMT3, or EFMT4. Furthermore, the E07 can neither read nor write using type JA, JJ, JR, or JW cartridges.

All 3592 tape drives support cartridge reuse. The 3592 tape cartridges can be reformatted to any tape format supported by the tape drive when writing from BOT. When reformatting, all existing data on the cartridge is erased. All compatible 3592 tape cartridge types can be used with the encryption function of the 3592 Models E05, E06, and E07. Encrypted cartridges use a unique format, Enterprise Encrypted Format 2 (EEFMT2), Enterprise Encrypted Format 3 (EEFMT3), or Enterprise Encrypted Format 4 (EEFMT4). Table 2-1 shows the IBM 3592 tape matrix.

Table 2-1 Types of IBM 3592 tape cartridges

<table>
<tr><th rowspan="2">Product, type, and SMS category</th><th colspan="4">Native capacity</th><th rowspan="2">Part number</th></tr>
<tr><th>E07</th><th>E06/EU6</th><th>E05</th><th>JA1</th></tr>
<tr><td rowspan="3">Data,
JA,
MEDIA5,
(610 m)</td><td>640 GB
(596.04 GiB)
E06 format</td><td>640 GB
(596.04 GiB)
E06 format</td><td rowspan="2">500 GB
(465.66 GiB)
E05 format</td><td rowspan="3">300 GB
(279.39 GiB)
(J1A format)</td><td rowspan="3">18P7534</td></tr>
<tr><td>500 GB
(465.66 GiB)
E05 format</td><td>500 GB
(465.66 GiB)
E05 format</td></tr>
<tr><td>300 GB
(279.39 GiB)
J1A format[1]</td><td>300 GB
(279.39 GiB)
J1A format</td><td>300 GB
(279.39 GiB)
J1A format</td></tr>
<tr><td rowspan="2">Extended
Data,
JB,
MEDIA9,
(825 m)</td><td>1600 GB
(1490.12 GiB)
E07 format</td><td rowspan="2">1000 GB
(931.32 GiB)</td><td rowspan="2">700 GB
(651.93 GiB)</td><td rowspan="2">Not supported</td><td rowspan="2">23R9830</td></tr>
<tr><td>1000 GB
(931.32 GiB)
E06 format</td></tr>
<tr><td>Advanced
Data,
JC,
MEDIA11,
(825 m)</td><td>4000 GB
(3725.29 GiB)
E07 format</td><td>Not supported</td><td>Not supported</td><td>Not supported</td><td>46X7452</td></tr>
</table>

<table>
<tr><th rowspan="2">Product, type, and SMS category</th><th colspan="4">Native capacity</th><th rowspan="2">Part number</th></tr>
<tr><th>E07</th><th>E06/EU6</th><th>E05</th><th>JA1</th></tr>
<tr><td rowspan="3">Economy,
JJ,
MEDIA7,
(246 m)</td><td>128 GB
(119.21 GiB)
E06 format[1]</td><td>128 GB
(119.21 GiB)
E06 format</td><td rowspan="2">100 GB
(93.13 GiB)
E05 format</td><td rowspan="3">60 GB
(58.88 GiB)
J1A format</td><td rowspan="3">24R0317</td></tr>
<tr><td>100 GB
(93.13 GiB)
E05 format</td><td>100 GB
(93.13 GiB)
E05 format</td></tr>
<tr><td>60 GB
(58.88 GiB)
J1A format[1]</td><td>60 GB
(58.88 GiB)
J1A format</td><td>60 GB
(58.88 GiB)
J1A format</td></tr>
<tr><td>Advanced
Economy,
JK,
MEDIA13,
(146 m)</td><td>500 GB
(465.66 GiB)
E07 format</td><td>Not supported</td><td>Not supported</td><td>Not supported</td><td>46X7453</td></tr>
<tr><td rowspan="3">Economy
WORM,
JR,
MEDIA8,
(246 m)</td><td>128 GB
(119.21 GiB)
E06 format[1]</td><td>128 GB
(119.21 GiB)
E06 format</td><td rowspan="2">100 GB
(93.13 GiB)
E05 format</td><td rowspan="3">60 GB
(58.88 GiB)
J1A format</td><td rowspan="3">24R0317</td></tr>
<tr><td>100 GB
(93.13 GiB)
E05 format</td><td>100 GB
(93.13 GiB)
E05 format</td></tr>
<tr><td>60 GB
(58.88 GiB)
J1A format</td><td>60 GB
(58.88 GiB)
J1A format</td><td>60 GB
(58.88 GiB)
J1A format</td></tr>
<tr><td rowspan="3">WORM,
JW,
MEDIA6,
(610 m)</td><td>640 GB
(596.04 GiB)
E06 format</td><td>640 GB
(596.04 GiB)
E06 format</td><td rowspan="2">500 GB
(465.66 GiB)
E05 format</td><td rowspan="3">300 GB
(279.39 GiB)
J1A format</td><td rowspan="3">18P7538</td></tr>
<tr><td>500 GB
(465.66 GiB)
E05 format</td><td>500 GB
(465.66 GiB)
E05 format</td></tr>
<tr><td>300 GB
(279.39 GiB)
J1A format[1]</td><td>300 GB
(279.39 GiB)
J1A format</td><td>300 GB
(279.39 GiB)
J1A format</td></tr>
<tr><td rowspan="2">Extended
WORM,
JX,
MEDIA10,
(825 m)</td><td>1600 GB
(1490.12 GiB)
E07 format</td><td rowspan="2">1000 GB
(931.32 GiB)
E06 format</td><td rowspan="2">700 GB
(651.93 GiB)</td><td rowspan="2">Not supported</td><td rowspan="2">23R9831</td></tr>
<tr><td>1000 GB
(931.32 GiB)
E06 format</td></tr>
<tr><td>Advanced
WORM,
JY,
MEDIA12,
(880 m)</td><td>4000 GB
(3725.29 GiB)
E07 format</td><td>Not supported</td><td>Not supported</td><td>Not supported</td><td>46X7454</td></tr>
<tr><td>Cleaning,
CLNxxxJA3</td><td>N/A</td><td>N/A</td><td>N/A</td><td>N/A</td><td>18P7535</td></tr>
</table>

Product, type, and SMS category	Native capacity				Part number
	E07	E06/EU6	E05	JA1	

Notes:
[1] The 3592-E07 supports reading JA, JJ, JR, and JW cartridges only with code level D3I3_5CD or later.

2.7.2 IBM 3592 Tape Cartridge read and write formats

This topic provides general information about the IBM 3592 tape cartridge.

The 3592 tape drive has a bidirectional read/write head capable of operating at four different recording densities, depending on the tape drive model. The 3592 E07 reads data in EFMT2 (896 tracks on 16 channels), EFMT3 (1152 tracks on 16 channels), and EFMT4 (2176 tracks on 32 channels) but only writes EFMT3 and EFMT4. The 3592 E06 reads data in EFMT1 (512 tracks on eight channels), EFMT2, and EFMT3, but only writes EFMT2 and EFMT3. The 3592 E05 drive reads and writes EFMT1 and EFMT2. The 3592 J1A reads and writes only EFMT1. Table 2-2 shows the new IBM 3592 recording format.

Table 2-2 Supported 3592 read (R) and write (W) formats

IBM 3592 Tape Drive	EFMT1 512 tracks/ 8 channels	EFMT2 896 tracks/ 16 channels	EFMT3 1152 tracks/ 16 channels	EFMT4 2176 tracks/ 32 channels
Model J1A	R/W	Not supported	Not supported	Not supported
Model E05	R/W	R/W[1]	Not supported	Not supported
Model E06[2]	R	R/W	R/W	Not supported
Model E07[2]	R[4]	R[4]	R/W[3]	R/W

[1] Model E05 can read and write EFMT1 operating in native or J1A emulation mode.
[2] Model E06 and E07 do not support emulation.
[3] Cartridge types JB and JX only.
[4] Model E07 can read JA, JJ, JR, and JW cartridge types only with code level D3I3_5CD or higher.

2.7.3 DFSMSrmm IBM media information

The Media type field specifies the type of physical media of the volumes defined to DFSMSrmm. If the media is not IBM media, the media type is based on the installation-defined media information.

The following list shows possible values for IBM media information:

*****	The volume is not a cartridge.
CST	Cartridge System Tape.
ECCST	Enhanced Capacity Cartridge System Tape.
EHPCT	Extended High Performance Cartridge Tape.
HPCT	High Performance Cartridge Tape.
MEDIA5/ETC	IBM Enterprise Tape Cartridge.
MEDIA6/EWTC	IBM Enterprise WORM Tape Cartridge 3592.
MEDIA7/EETC	IBM Enterprise Economy Tape Cartridge 3592.
MEDIA8/EEWTC	IBM Enterprise Economy WORM Tape Cartridge 3592.
MEDIA9/EXTC	IBM Enterprise Extended Tape Cartridge 3592.

MEDIA10/EXWTC	IBM Enterprise Extended WORM Tape Cartridge 3592.
MEDIA11/EATC	IBM Enterprise Advanced Tape Cartridge.
MEDIA12/EAWTC	IBM Enterprise Advanced WORM Tape Cartridge.
MEDIA13/EAETC	IBM Enterprise Advanced Economy Tape Cartridge.

2.7.4 IBM 3592 cartridge memory

This topic describes the *cartridge memory* (CM) of the 3592 tape cartridge.

Each 3592 data cartridge contains a passive, contactless, silicon storage device called a CM. This CM holds information about the cartridge and the media in the cartridge, and holds statistics about the media in the cartridge. The cartridge and media information is stored in a protected, read-only area of the CM. This information is read by the CM reader in the drive, by using a contactless, radio-frequency interface when the cartridge is loaded into the drive. The media performance statistics are stored in an unprotected, read/write area of the CM. These statistics are updated by the CM reader just before the cartridge is unloaded. The media performance statistics are maintained by the Statistical Analysis and Reporting System (SARS) portion of the drive microcode. Each cleaning cartridge also contains a CM, which tracks the number of cleaning uses.

2.8 Volume management

DFSMSrmm manages volumes based on the volume type, volume status, and the volume media.

2.8.1 Types of volumes

DFSMSrmm provides support for physical, logical, and stacked volumes (see Figure 2-21).

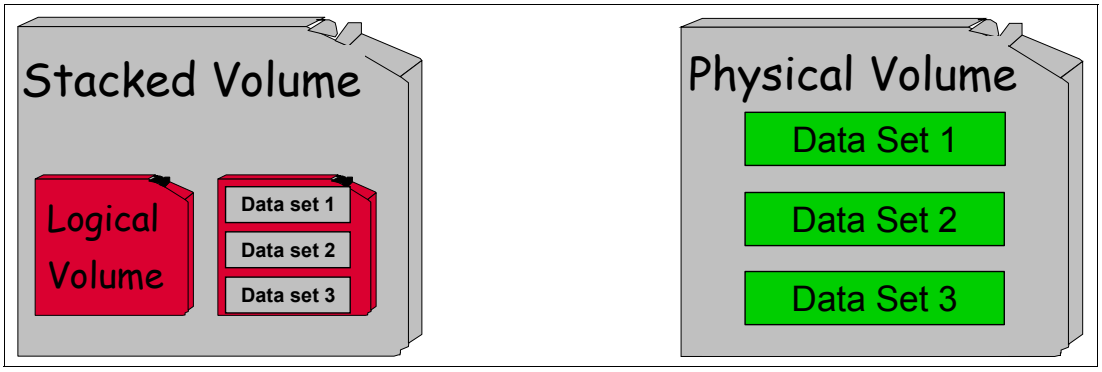


Figure 2-21 Stacked, logical, and physical volumes

Physical volumes

DFSMSrmm provides support for all types of physical volumes, tracking their use and the data they contain, and providing vaulting and retention services according to policies assigned to the data sets on the volumes.

Logical volumes

These volumes reside in a VTS or on exported stacked volumes. DFSMSrmm tracks the use of logical volumes and the data that resides on the volumes. Retention services are provided according to policies that are assigned to the data sets on the volumes. DFSMSrmm supports exporting of logical volumes on stacked volumes and provides vaulting services for these stacked volumes based on the policies assigned to the data sets on the contained logical volumes. Logical volumes can be removed from a VTS by using export processing.

Stacked volumes

These volumes have a one-to-one association with physical tape media and are used in a VTS to store logical volumes. Stacked volumes are not used by MVS applications. They are used by the VTS and its associated utilities. Stacked volumes can be removed from a VTS to allow transportation of logical volumes to a vault or to another VTS.

DFSMSrmm has added support for stacked volumes to allow identification of the stacked volumes in a VTS and direct management of these volumes when exported. These stacked volumes are assigned to specific slots when moved to storage locations that require shelf management.

DFSMSrmm provides support for importing logical volumes on stacked volumes either into the VTS from which it was exported, or into a different VTS.

If stacked volume support is enabled, you can define stacked volumes and the logical volumes associated with them by using the RMM ADDVOLUME subcommand as shown in Figure 2-22.

```
RMM AV convol TYPE(STACKED) LOCATION(SHELF)
RMM AV volser TYPE(LOGICAL/PHYSICAL) STATUS(MASTER) CONTAINER(convol)
```

Figure 2-22 Adding a stacked volume as a container volume

You also can use stacked volume support as a way to manage the boxes in which you transport and store tapes. You can associate volumes to a box, by defining the box as a stacked volume, and using RMM TSO subcommands to add the volumes to the box as the CONTAINER.

2.8.2 Volume status

Within DFSMSrmm, a volume can be in one of six different statuses:

Scratch This status indicates that the volume is available for use. It is a volume that does not contain valid data.

Master This status is assigned to a volume at open time after a non-specific mount.

User This is the status that is assigned to a volume after a user gets a volume request or it is assigned through an DFSMSrmm command.

Pending release This is a transient status that occurs after a master or user volume expires and before becoming a scratch volume.

Init This volume is a scratch volume that is waiting to be initialized. When it is initialized, the status changes to SCRATCH.

Entry The volume is a scratch volume coming into an automatic tape library (ATL). When entered, the status changes to SCRATCH or INIT.

Figure 2-23 shows the lifecycle of a volume.

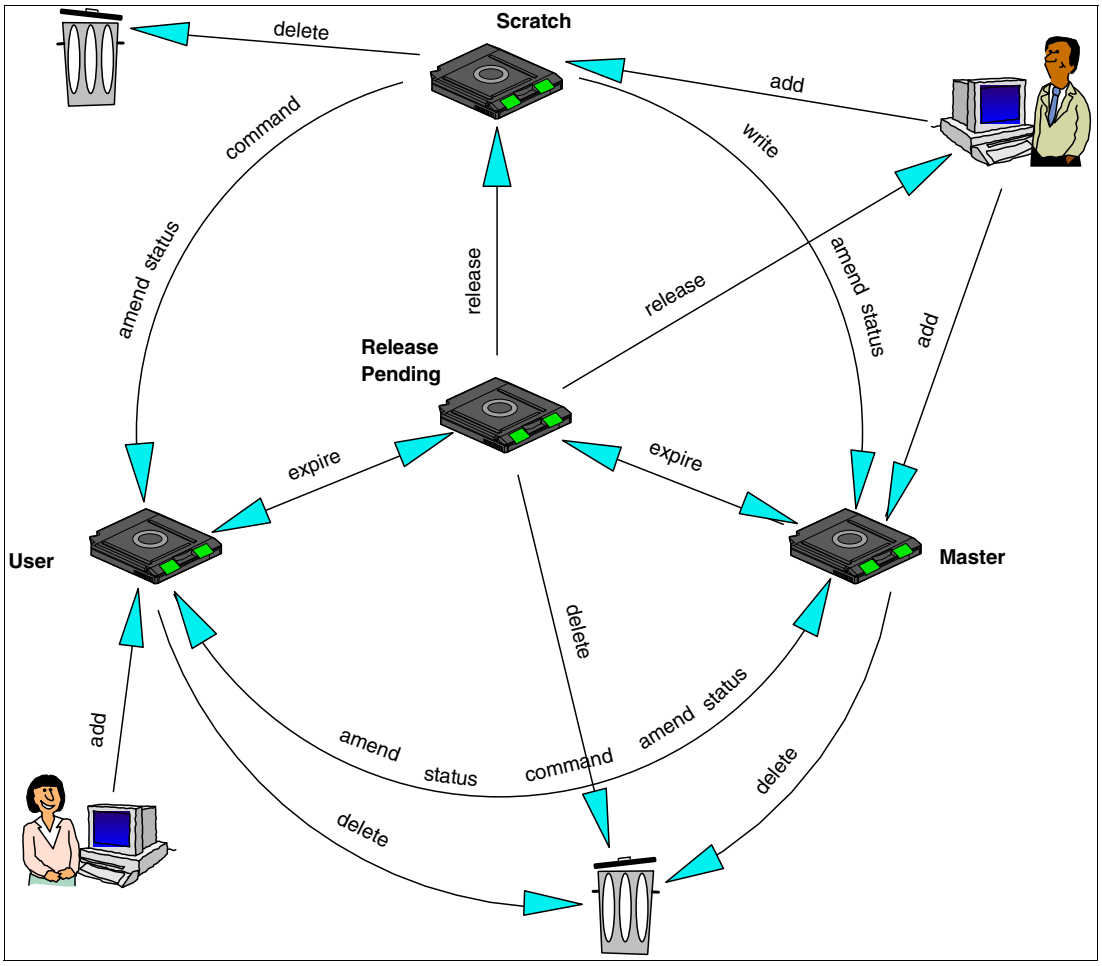


Figure 2-23 Lifecycle of a volume

Most of the volumes that DFSMSrmm manages in your installation are probably *scratch* volumes, that is, volumes that are used again and again by different users. Each time a volume is used, it is retained and managed by policies that you define to DFSMSrmm.

When the data is no longer required, the volume is returned to scratch status, and is ready for use by another user.

With DFSMSrmm, you can add or delete volumes without stopping your normal tape activities.

In addition to managing scratch volumes, DFSMSrmm can manage any volume you define to it, including those that you treat as *foreign volumes* under your existing system. DFSMSrmm supports any volume serial (VOLSER). You can still process volumes as foreign to DFSMSrmm by either not defining them to DFSMSrmm or using the DFSMSrmm *ignore* support through the EDGUX100 exit.

DFSMSrmm records information for data sets on all files of a tape volume. It can also manage the RACF TAPEVOL and TAPEDSN profiles for tape volumes and uncataloged data sets. Figure 2-24 on page 51 shows how DFSMSrmm processes records related to tape data sets.

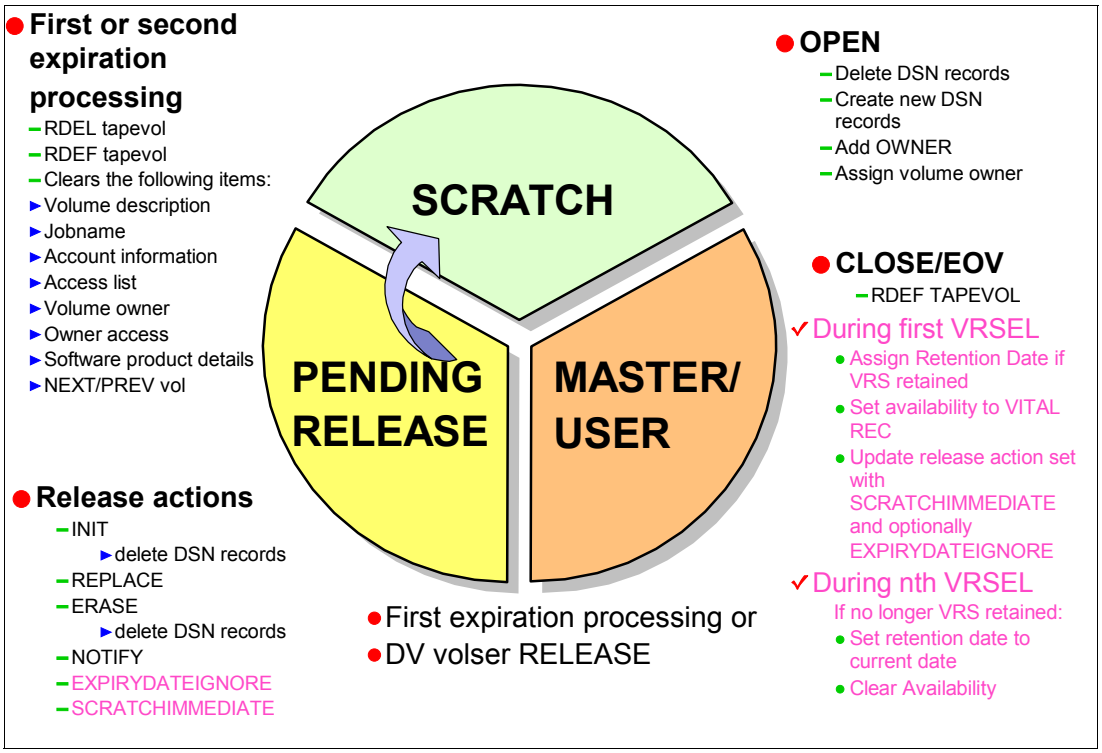


Figure 2-24 Timing of related events for a volume

Note: If you have specified the “SCRATCHIMMEDIATE” keyword in your VRS definitions, the status is directly changed from MASTER/USER to SCRATCH without a requirement to run a second expiration processing.

If you have installed DFSMSrmm z/OS V1R13 or higher and you have specified the retention method RETENTIONMETHOD(EXPDT) in the EDGRMMnn parmlib member, the DFSMSrmm inventory management EXPROC processing always attempts to return volumes to scratch on the same run as the volume is released (this is as if the SCRATCHIMMEDIATE attribute is set for the volume). DFSMSrmm maintains a consistent expiration date and time for all data set records of a multi-volume data set.

2.8.3 Validation

DFSMSrmm automatically validates volumes, ensuring that only valid scratch volumes are mounted for nonspecific mount requests and that the correct volume is mounted for a specific mount request. This validation eliminates the unintentional overwriting of a valid master volume, or a volume retained for disaster recovery or vital record management.

When a data set on a volume is opened and closed, DFSMSrmm automatically:

- Changes the volume status from scratch to master for nonspecific mount requests
- Sets an expiration date for the volume and ensures that the maximum expiration date is not exceeded (repeated at close time) as defined in the PARMLIB member
- Records information about data sets on the volume (the data set name is recorded at open time; all other information is recorded at close time)
- Counts the number of times a volume is used

- Counts the number of temporary and permanent errors encountered
- Sets a security classification (as defined in the DFSMSrmm PARMLIB) based on the data sets that reside on the volume
- Prevents reading of data on a volume in scratch status when DFSMSrmm is running in protect mode

All of these actions occur at open time, except where noted.

2.8.4 Ignore volume support

DFSMSrmm can ignore volumes regardless of whether the volumes were defined to DFSMSrmm. To request that DFSMSrmm ignore a volume, tailor the EDGUX100 DFSMSrmm installation exit or specify an OPENRULE definition in EDGRMMnn PARMLIB to use undefined volumes or duplicate volumes.

Using OPENRULE and PRTITION to ignore volume serial numbers

The OPENRULE and PRTITION commands allow you to define whether they apply to volumes defined to DFSMSrmm or not. You can use operands on the OPENRULE command to automatically ignore volumes, rather than using EXPDT=98000, ACCODE=xCANORES, or a customized EDGUX100.

To prevent DFSMSrmm from rejecting undefined or foreign volumes, use an action of IGNORE to request that DFSMSrmm ignore the undefined volumes. All volumes that are ignored with the IGNORE action must be authorized to be ignored. You can use OPENRULE for any set of volumes identified with the VOLUME and VOLUMERANGE operands.

To request that DFSMSrmm ignore a volume, you also have to define a RACF FACILITY class entity as shown in Figure 2-25.

For a detailed description of how to use the OPENRULE and PRTITION commands, see 8.5, “Using PRTITION and OPENRULE commands” on page 290.

Note: OPENRULE is the preferred command for rejecting and ignoring volumes. The previously used REJECT command and EDGUX100 exit module will continue to work if they have already been specified.

Using EDGUX100 to ignore volume serial numbers

When you use the EDGUX100 installation exit, DFSMSrmm calls the exit each time a volume is opened. The sample installation exit checks the JCL-specified EXPDT value for the special date 98000 or for the ACCODE value *x*CANORES. If the 98000 special date or the ACCODE value *x*CANORES is found, the EDGUX100 exit uses the installation exit parameter list to request that DFSMSrmm ignore the volume. Figure 2-27 on page 53 shows how the EDGUX100 user exit works. This support was introduced with APAR OW53763.

Bypass DFSMSrmm processing

To request that DFSMSrmm ignore a volume, you also have to define a RACF FACILITY class entity as shown in Figure 2-25.

```
STGADMIN.EDG.IGNORE.TAPE.VOLSER
```

Figure 2-25 Global resource to ignore tape volumes

If you want to distinguish between volumes that are managed by DFSMSrmm and volumes that are not managed by DFSMSrmm, define the RACF profiles as shown in Figure 2-26 to check whether the user is authorized to request that DFSMSrmm ignore the volume.

```
STGADMIN.EDG.IGNORE.TAPE.RMM.VOLSER
STGADMIN.EDG.IGNORE.TAPE.NORMM.VOLSER
```

Figure 2-26 More specific resources to ignore tape volumes

In the RACF profile, *VOLSER* is the volume serial number of the mounted volume or requested tape volume. Figure 2-27 shows how the BYPASS DFSMSrmm function works.

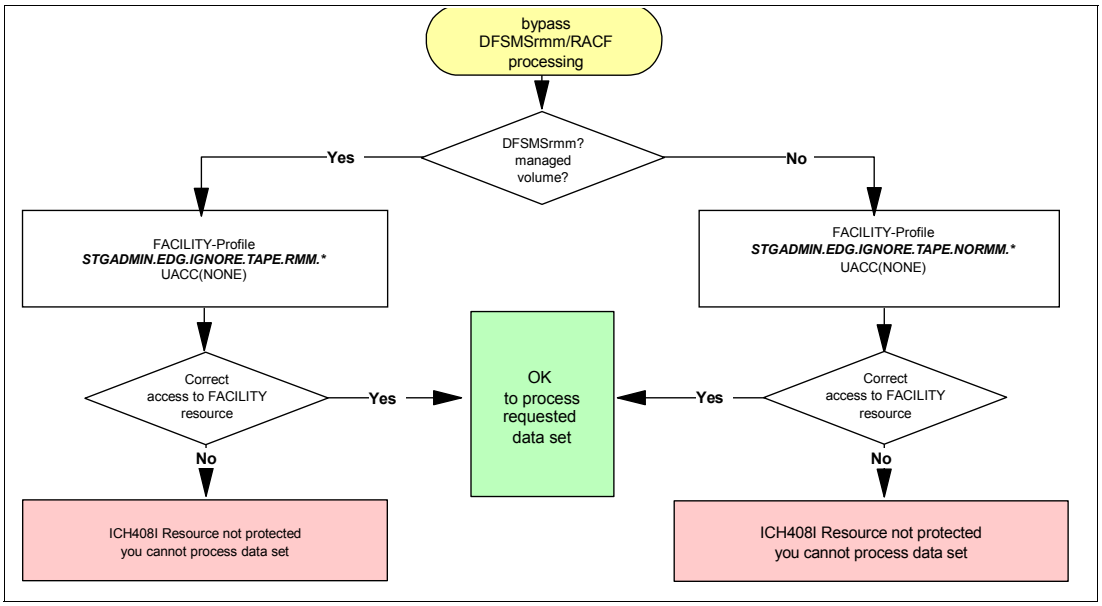


Figure 2-27 Bypass DFSMSrmm function

Note: When the profiles STGADMIN.EDG.IGNORE.TAPE.**RMM**.*VOLSER* and STGADMIN.EDG.IGNORE.TAPE.**NORMM**.*VOLSER* are not defined, DFSMSrmm looks for STGADMIN.EDG.IGNORE.TAPE.*VOLSER*.

If you define the *RMM* and *NORMM* ignore profiles, DFSMSrmm makes the authorization checks as shown in Table 2-3.

Table 2-3 DFSMSrmm ignore profile checking

Authorization check	Profile used
Volser not found in CDS	STGADMIN.EDG.IGNORE.TAPE.**NORMM**.*VOLSER*
Volser found in CDS and matches on HDR1 17-character data set name	STGADMIN.EDG.IGNORE.TAPE.**RMM**.*VOLSER*
Volser found in CDS and no match on HDR1 17-character data set name	STGADMIN.EDG.IGNORE.TAPE.**NORMM**.*VOLSER*

2.8.5 Support for duplicate volumes

In DFSMSrmm, volume serial numbers are used to identify volumes and to identify the volume label. DFSMSrmm allows you to define a volume using a serial number that is different from the one that is recorded in the volume label. This way you can define volumes with duplicate volume serial numbers.

DFSMSrmm open processing has been updated to detect duplicate volumes and to use the correct CDS volume information to manage the volume. You can continue to have duplicate volumes ignored, but the suggestion is to add them to the DFSMSrmm CDS and manage them. Figure 2-28 shows how DFSMSrmm supports duplicate volume serial numbers by specifying VOLSER, RACK, and the new VOL1 operand of the ADDVOLUME function.

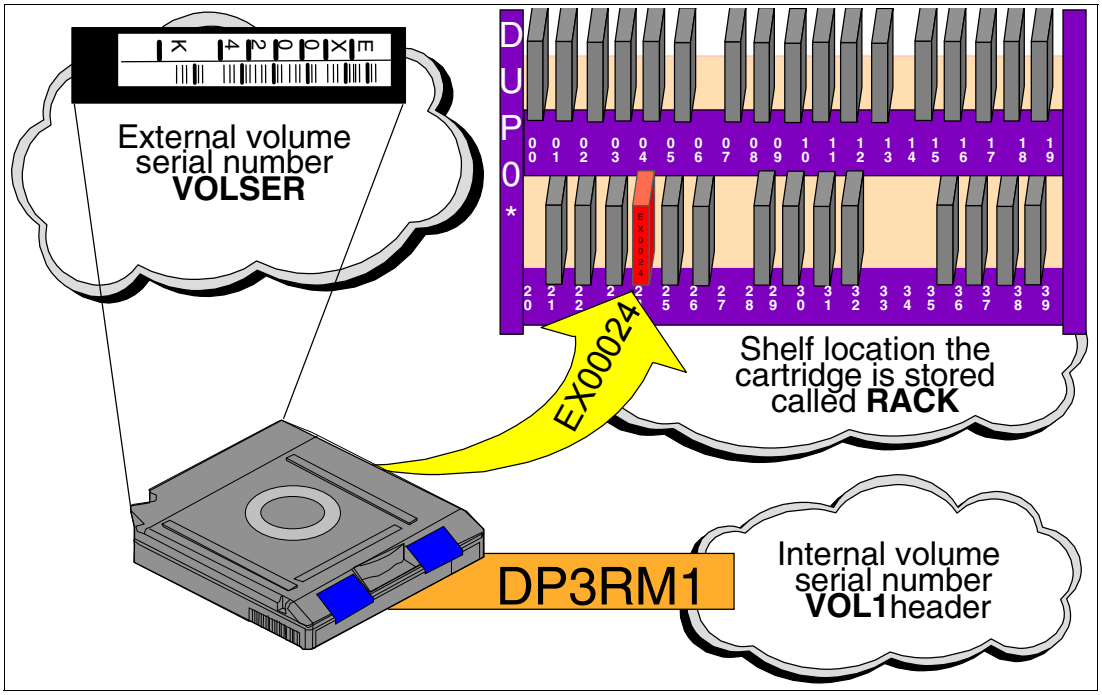


Figure 2-28 DFSMSrmm duplicate volume support

Each volume that is defined to DFSMSrmm must be defined using a unique volume serial number, but for private physical volumes you can also define the VOL1 label VOLSER. You can have any number of duplicate volumes with the same VOL1 label VOLSER. You can use duplicate volumes for both input and output processing, and do not need to use BLP or DFSMSrmm ignore processing.

You do not need to enable duplicate volume support. You simply start to define and use duplicate volumes when you need the support. You use the DFSMSrmm ADDVOLUME or CHANGEVOLUME subcommand with VOL1(*VOLSER*) to define a duplicate volume. See Figure 2-29 on page 55.

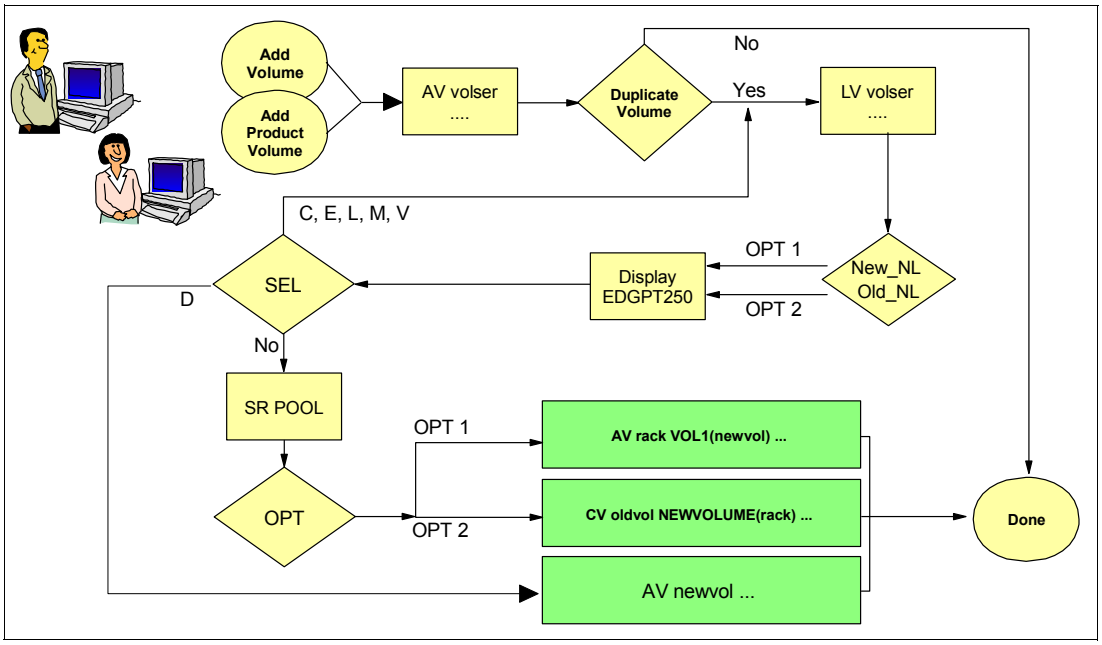


Figure 2-29 Adding a duplicate volume serial number

A no-label volume cannot be a duplicate volume, because you can define such a volume to DFSMSrmm using any unique value for the VOLSER. Also, no special processing is required by DFSMSrmm to distinguish the volume from another identical volume.

DFSMSrmm is the only component in the system that knows a volume is a duplicate. DFSMSrmm hides this fact from others by substituting the VOL1 label VOLSER read from the volume with the requested volume when DFSMSrmm has validated that the correct volume is mounted. The requested volume is the unique volume serial number that you specify in your JCL or allocation request that identifies the volume to DFSMSrmm.

DFSMSrmm exploits the capabilities of the Tape Mount exit to check the VOL1 label VOLSER. DFSMSrmm uses the requested VOLSER to find a volume defined to RMM, and validates that the VOL1 label VOLSER matches the VOL1 label VOLSER defined to DFSMSrmm. The user-specified VOLSER identifies the volume record to be read from the DFSMSrmm CDS. The VOL1 label VOLSER must match the DFSMSrmm-recorded VOL1 label VOLSER, or the volume record has no recorded VOL1 label VOLSER, and the requested volume matches the VOL1 label VOLSER.

Note: In DFSMSrmm, there are three different values to identify a volume:

VOLSER	To specify the volume serial number
VOL1	To define a duplicate volume when the VOL1 label volume serial number does not match the volume you are defining to DFSMSrmm
RACK	Specifies a shelf location in the removable media library where DFSMSrmm stores the volume

If you add a new volume with a volume serial number of a volume previously defined in the CDS, DFSMSrmm displays a new panel as shown in Figure 2-30 on page 56.

```
                        DFSMSrmm Add Duplicate Volume
Command ===>                                                  Scroll ===> CSR

Volume VT0002 duplicates existing volume VT0002
  Enter selected option for volume VT0002 or use line command below
  Any recommended option is pre-selected for you

Option ===>
  1 Add the volume as a duplicate
  2 Change the existing volume to be a duplicate

Specify a pool for the duplicate volume
Pool prefix ===>

Enter HELP or PF1 for the list of available line commands
   Volume VOL1   Rack                     Data        Product            Feat
S  serial volser number Status   Location Sets  Label number   Level     code
-- ------ ------ ------ -------- -------- ----- ----- -------- --------- ----
   VT0002               MASTER   VTFM001  1     SL
```

Figure 2-30 Adding a duplicate volume panel

2.9 Policies for retention and movement

DFSMSrmm provides policy management for movement and retention at the data set, volume, or volume set level. Every tape data set can have a policy, and each policy can specify movement as well as retention. The retention and movement policies you define to DFSMSrmm are known as *vital record specifications* (VRSs). You use them to indicate how long and where you want to keep data sets or volumes. You also use them to define how volumes are to be moved among the libraries that DFSMSrmm supports and the storage locations defined for vital records and disaster recovery purposes.

You can create a VRS *chain* to cause a sequence of volume moves. The first VRS in the chain is a data set name, SMS management class, DFSMSrmm management value, or volume VRS. These specify one location in which the data set or volume being retained is to be stored. You add one name VRS for each additional location or retention value that applies to the volume or data set.

DFSMSrmm records the starting location for a volume when the volume is initially defined to DFSMSrmm, or when the volume information is changed. This starting location is known to DFSMSrmm as a *home location*. Home is the place from which volumes start, and to which they are returned, when the identified retention and movement actions have been completed.

2.9.1 Data set retention

You use *data set names* and *data set name masks* to define retention policies for data sets, and you use *job names* and *job name masks* to define retention policies to further qualify the criteria for applying retention and movement policies.

For data sets, you can request the following types of retention:

- Retention by PARMLIB option - VRSEL(NEW)

  You can control the extent to which data set name and management value VRSs are used. You can define different retention criteria in a next VRS only if you specify VRSEL(NEW).

  Note: The VRSEL option OLD is no longer supported.

- Retention by cycles

  You can retain a minimum number of cycles or copies of a data set. This type of retention applies to generation data groups (GDGs) or like-named data sets identified by pseudo-GDG data set names. For non-GDG data sets, DFSMSrmm considers each occurrence of a data set to be a cycle.

- Retention by BYDAYSCYCLE

  You can specify to retain all instances of a data set created on a single day as a single cycle.

- Retention until expired

  You can retain a data set until the volume expiration date has been reached or the retention amount has elapsed.

- Retention by number of elapsed days

  You can retain each copy of a data set a given number of days since it was created.

- Retention by days since last referenced

  You can retain each copy of a data set for a set number of days since the data set was read or written.

- Retention while data set is cataloged

  You can retain any data set as long as it remains cataloged. Catalog status can also be used in combination with management by cycles, elapsed days, days since last referenced, and volume expiration date.

- Retention by extra days

  On a NEXTVRS, you can specify the number of days since a name VRS started to retain the data set. The number of extra days is specified with the COUNT operand. EXTRADAYS is the number of days since the NAME VRS started to retain the data set. The number of days depends on when the previous VRS stopped retaining the data set and the time when vital record processing is run. This option can be used only on a NEXTVRS and not a primary VRS.

- Retention to a specific date

  You can set a deletion date for a VRS. When that data is reached, the VRS is deleted. All data sets and volumes that match the VRS become eligible for release processing, or they might match a less specific VRS that specifies different retention and movement information.

- Retention by expiration date

  You can retain the data set on a volume as long as the volume expiration date has not yet been reached. You can also use a combination of catalog status, volume expiration date, and other retention options to retain data sets.

- Retention of open data sets

  You can specify a separate policy to apply to all data sets that are currently open.

- Retention of data set closed by abend processing

  You can specify a separate policy to apply to all data sets that were open at the time of an application or system abend.

- Retention of data sets closed with normal disposition DELETE

  You can specify a separate policy to apply to all data sets that were created with normal disposition of DELETE.

- Retention of set

  This allows you to retain all volumes of a set until all of the data sets on those volumes have expired. Use the EDGRMMxx PARMLIB option RETAINBY(SET).

Figure 2-31 shows an example of a multifile/multivolume set. The first and second data sets are not expired, but the third one has expired since yesterday. The third volume of the set is only retained if PARMLIB option RETAINBY(SET) is specified.

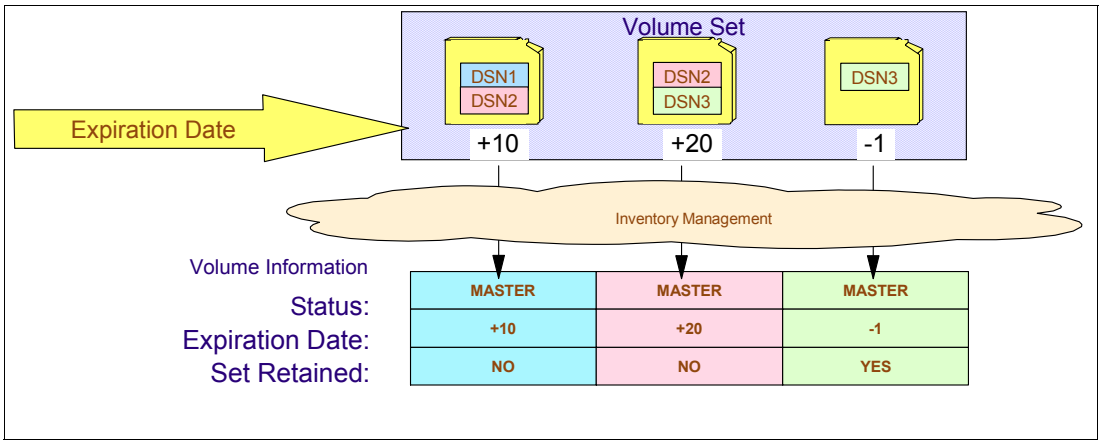


Figure 2-31 RETAINBY(SET): Expiration date

Your installation can also define a VRS management value or an SMS management class to define retention and movement policies. If you have specified VRSEL(NEW) in the EDGRMMxx PARMLIB member, you can have up to two policies per data set: one policy using a data set name mask, and one using a VRS management class or VRS management value.

A VRS management value assigns management and retention values to tape data sets. You can define data set VRSs for VRS management values to provide support for special JCL-specified expiration dates, and to allow DFSMSrmm to manage those data sets with special dates. You can use the sample installation exit EDGUX100, in SAMPLIB, to assign VRS management values.

2.9.2 Volume retention

To define retention and movement policies for volumes, you can use a specific or generic VOLSER. If you use a specific VOLSER, the volume matching that number is retained by days since creation. If you use a generic VOLSER, any volumes matching the generic VOLSER are retained. These volumes must be of user or master status.

For a volume, you can request the following types of retention using a specific or generic VOLSER:

- Retention by number of elapsed days since creation

  Indicates that a volume be retained until the specified number of days since creation have elapsed

- Retention by number of volumes

  Indicates how many volumes need to be retained

2.9.3 Movement

DFSMSrmm records the starting location for a volume when the volume is initially defined to DFSMSrmm, or through a change to the volume information. This starting location is known to DFSMSrmm as a *home location*. Volumes are returned to the home location when the identified retention and movement actions have been completed.

You can give any system-managed library or storage location as a target destination for a volume move. Obviously, this location must be defined to DFSMSrmm. The location can be an ATL, MTL, storage location, or shelf.

You can also define policies to provide retention information for data sets and volumes that must be moved through multiple locations before they expire. You define such policies by creating a VRS chain.

When multiple data sets on the same volume are retained by VRSs, and each VRS contains a different destination, DFSMSrmm decides where to move the volume according to priority number.

Unlike some other systems, DFSMSrmm does not perform storage location management on the basis of the first file on a volume. Storage location management is based on the priority of the storage location management requirements for all data sets on a volume. Therefore, DFSMSrmm must sometimes perform conflict resolution, and only retains and moves volumes with a data set that is retained by a VRS.

A recently added function of DFSMSrmm allows you to move a SET of volumes, therefore keeping related volumes together. Use the PARMLIB option MOVEBY(SET).

Storage location management with DFSMSrmm is essentially similar to vault management in other tape management systems. What seems to be unique to DFSMSrmm is the concept of volume-in-transit. Volumes have been scheduled for a move or are on their way, but they have not yet been received at the storage location. Operators can tell which volumes are still in transit. A special command moves volumes from in-transit status to storage-location-resident status. Installations that have established special storage location names to cover transport time need fewer storage locations. To prevent librarians from issuing this command, automate the confirmation of movement by issuing TSO commands or with REXX procedures.

If you specified VRSEL(NEW) in the EDGRMMxx member of PARMLIB, you can now have separate movement and retention policies. Use a VRS that is based on a management value or management class name to define retention criteria; use a VRS that is based on a data set name mask for defining movement.

2.10 SMS ACS support

You can use the ACS routines to process the special calls that DFSMSrmm makes to the SMS subsystem for ACS processing to set an SMS management class. DFSMSrmm requests that the management class and storage group routines are run. The environment variable is set to RMMPOOL, so that you can differentiate allocation requests for system-managed data sets from requests by DFSMSrmm for a storage group.

You can use SMS ACS to decide whether data sets are to be system-managed or non-system-managed. Using the SMS pre-ACS calls to DFSMSrmm helps you to use your SMS ACS logic. Use the pre-ACS processing for decisions about the MSPOOL and MSPOLICY values that come from DFSMSrmm scratch pool and EDGUX100 exit processing. You can plan to move the EDGUX100 exit decisions into your SMS ACS routines to enable the EDGUX100 exit processing to be ignored or removed some time in the future.

DFSMSrmm attempts to pass values for the *MSPOOL* and *MSPOLICY* ACS read-only variables when the PARMLIB option PREACS(YES) is set. If you do not use the EDGUX100 exit, the *MSPOOL* variable is set to the DFSMSrmm system-based scratch pool decision. If you do not use the EDGUX100 exit, there is never a value set for *MSPOLICY*. DFSMSrmm sets a pre-ACS variable only if the variable has not yet been set using the pre-ACS exit. Because DFSMSrmm gets control after the installation exit, any vendor or client decisions take precedence (see Figure 2-32).

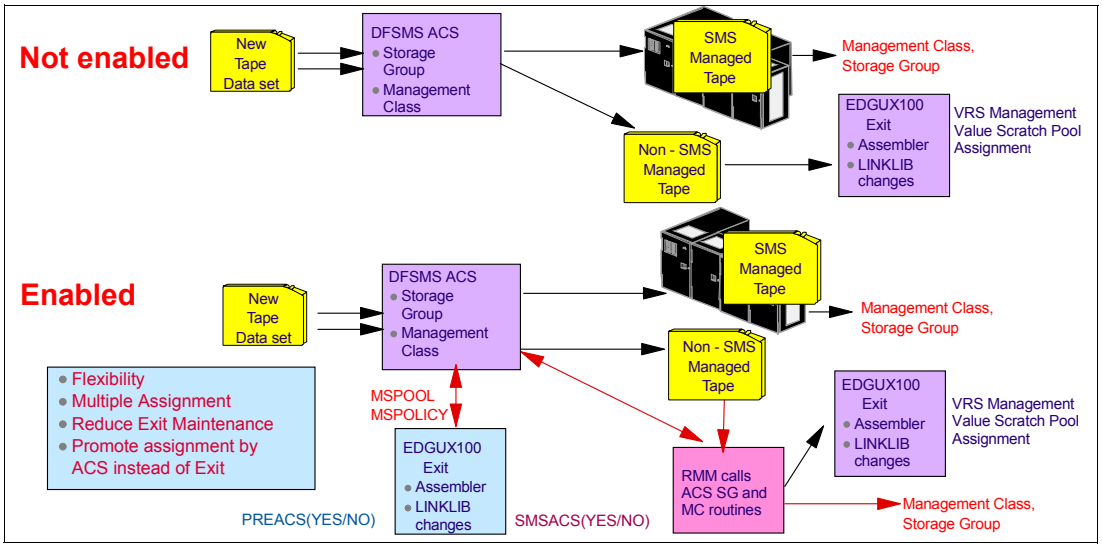


Figure 2-32 DFSMSrmm and SMS ACS support

2.11 DFSMSrmm automated tape library support

DFSMSrmm excels at ATL management. In an ATL environment, you have different databases to store volume and location information. In the DFSMSrmm CDS, you have the volume serial number, the volume status, and the volume location information. In the SMS tape volume database or tape configuration database (TCDB), you have the volume serial number, the volume status, and the location information. In the library manager database, you have the volume serial number and the volume status information.

DFSMSrmm keeps all three databases in sync by using all four available OAM installation exits:

CBRUXCUA	Change use attribute exit
CBRUXEJC	Eject cartridge exit
CBRUXENT	Entry cartridge exit
CBRUXVNL	Volume not in library exit

For more details, see the “Running DFSMSrmm with System-Managed Tape Libraries” chapter in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405. Figure 2-33 shows the entire ATL environment.

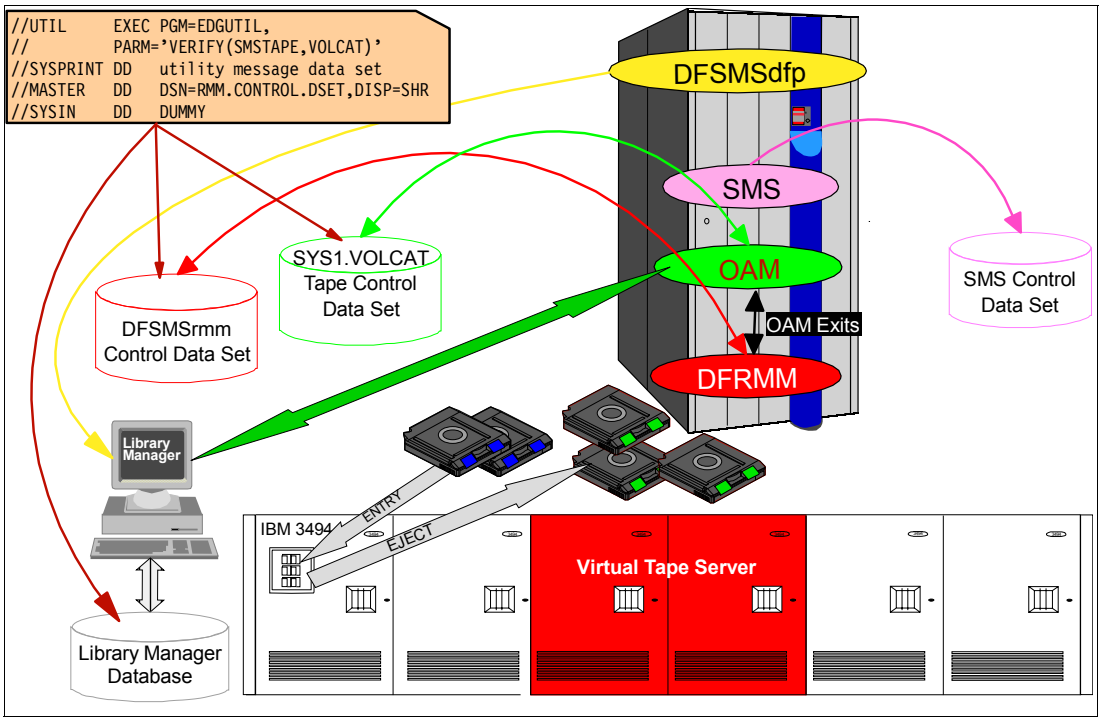


Figure 2-33 ATL environment

The system-managed tape has the following components:

- OAM and its Library Control System (LCS) component
- Storage management subsystem (SMS)
- Integrated Storage Management Facility (ISMF)
- Installation-wide exits
- Tape management system
- Library Manager

DFSMSrmm provides the following support for system-managed tape libraries:

- Cartridge entry processing
- Cartridge eject processing
- Expiration management
- Volume-not-in-library support
- Movement between libraries and storage locations
- Partitioning the libraries with other systems
- VTS import/export processing
- Tracking and managing logical, physical, and stacked volumes

The following sections review some of the items in the list.

2.11.1 Cartridge entry processing

DFSMSrmm automatically adds volumes to the DFSMSrmm control data set during entry processing and when you use volumes. If a volume that is defined to DFSMSrmm is entered into a system-managed tape library without using the RMM CHANGEVOLUME command to set the volume location to the library name, DFSMSrmm updates the DFSMSrmm control data set volume record with the library name, library type, and volume entry status. DFSMSrmm does not update the home location name for the volume.

DFSMSrmm allows you to partition system-managed tape libraries using the DFSMSrmm REJECT PARMLIB option to identify volumes that you want to partition. Volumes can be identified based on volume prefix and as individual volumes defined to DFSMSrmm on z/OS.

The REJECT parameter of the EDGRMMxx PARMLIB can be used to control cartridge entry processing into the library and whether the cartridge is to be used for input processing only. Figure 2-34 shows examples of using the REJECT command in the EDGRMMxx member.

```
REJECT OUTPUT(CC12*)
REJECT ANYUSE(VM1*),OUTPUT(VM*)
REJECT ANYUSE(DD0*)
REJECT ANYUSE(*)
```

Figure 2-34 Old REJECT definitions

Important: Define all volumes to DFSMSrmm before entering them into a library. Define private volumes before their entry into the automated tape library, and set the ISMF default cartridge entry status to scratch.

Use the new PRTITION and OPENRULE parameters to replace the REJECT parameter of the EDGRMMxx PARMLIB definitions. Use the OPENRULE command to identify special processing for DFSMSrmm to perform during OPEN processing. It applies equally to both system-managed and non-system-managed volumes. Use the PRTITION command to identify special processing for DFSMSrmm to perform during library entry, export/import, eject, and Common User Access (CUA) processing, during OPEN processing, and EXPROC processing. The PRTITION commands apply to all volumes, not only system-managed volumes. However, only system-managed volumes benefit for functions such as library insert, CUA, and eject.

Note: Although the REJECT command is still supported by DFSMSrmm, use the OPENRULE and PRTITION commands to prevent a range of tapes from being used on a particular system. If you are already using REJECT commands, consider converting them to OPENRULE and PRTITION commands.

Figure 2-35 on page 63 shows examples of using the OPENRULE and PRTITION commands in the EDGRMMxx member.

```
         /* Allow read of undefined selected volumes                    */
OPENRULE VOLUME(CC12*) TYPE(RMM) OUTPUT(REJECT) INPUT(ACCEPT)
OPENRULE VOLUME(VM*) TYPE(RMM) OUTPUT(REJECT) INPUT(ACCEPT)
         /* Ignore for input selected system managed volumes if authorized */
OPENRULE VOLUME(DDO*) TYPE(NORMM)                                         -
         INPUT(IGNORE) OUTPUT(REJECT)
OPENRULE VOLUME(VM1*) TYPE(NORMM)                                         -
         INPUT(IGNORE) OUTPUT(REJECT)
         /* Global partition rule - ignore all smt and nosmt volumes     */
PRTITION VOLUME(*) TYPE(RMM) -
         SMT(IGNORE) NOSMT(IGNORE)
         /* This partition owns JT* volumes and the volume is added to   */
         /*      the DFSMSrmm CDS if not yet defined                     */
PRTITION VOLUME(JT*) TYPE(ALL) SMT(ACCEPT)
```

Figure 2-35 New OPENRULE and PRTITION definitions

Important: Define all volumes to DFSMSrmm before entering them into a library. Define private volumes before entry into the automated tape library, and set the ISMF default cartridge entry status to scratch.

2.11.2 Ejecting volumes from system-managed libraries

You can eject physical volumes from a system-managed library by using various methods:

- Using the library manager console commands
- Using ISMF commands
- Using DFSMSrmm commands through ISPF or batch

2.11.3 Volume-not-in-library processing

When a volume is requested for a job, and is not in a system-managed tape library, the DFSMSrmm supplied CBRUXVNL exit is called so the volume can be inserted into a library to prevent job failures from occurring.

2.11.4 Verifying the databases

You can use the DFSMSrmm utility EDGUTIL with the parameters VERIFY(VOLCAT,SMSTAPE) to check the consistency of information in the DFSMSrmm control data set, the tape configuration database (TCDB), and the library manager database. If you have differences, you can use the same utility with a PARM of MEND to correct the errors. Figure 2-36 on page 64 shows the entire environment if you have an IBM ATL installed.

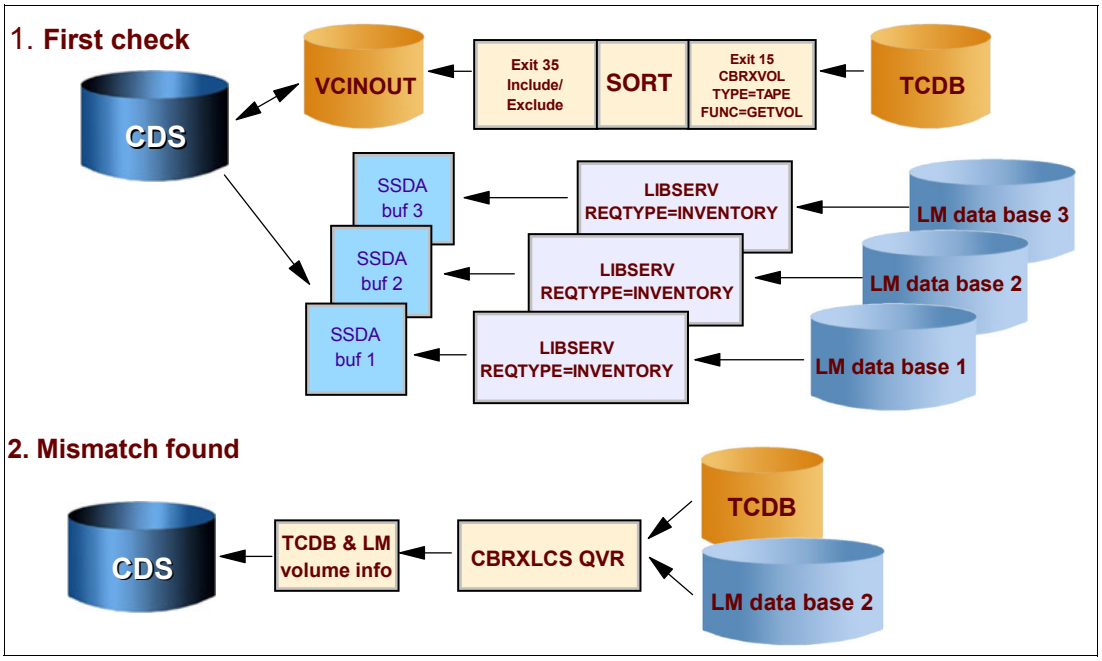


Figure 2-36 EDGUTIL utility to verify the VOLCAT and LM database

Important: To correct discrepancies, you can use the MEND function, but client knowledge is necessary.

2.11.5 VTS import/export processing

DFSMSrmm supports VTS libraries, such as automated system-managed libraries, with extensions to handle the special requirements for different volume types and functional capability. DFSMSrmm supports VTS import/export in different ways depending on whether DFSMSrmm stacked volume support is enabled. When you enable stacked volume support, DFSMSrmm tracks logical volumes and stacked volumes.

When you do not enable stacked volume support, DFSMSrmm does not track the stacked volumes. DFSMSrmm supports the IBM Peer-to-Peer VTS by ensuring that you cannot use the names of distributed VTS libraries with DFSMSrmm. You must only use the names of consolidated libraries with DFSMSrmm.

2.11.6 Defining stacked volumes to DFSMSrmm

You can define stacked volumes whether stacked volume support is enabled. DFSMSrmm defines the stacked volumes automatically at export time when stacked volume support is enabled. There is no automatic host notification when stacked volumes enter the VTS, leave the VTS, or change category, so you must use DFSMSrmm commands if you want to have all stacked volumes defined to DFSMSrmm.

Note: Always define stacked volumes to DFSMSrmm. This ensures that the volumes cannot be used outside of the VTS. It also ensures that DFSMSrmm checks at cartridge entry processing time that a physical volume or logical volume does not duplicate a stacked volume.

2.12 Catalog synchronization

When you run DFSMSrmm with user catalogs and the DFSMSrmm control data set unsynchronized, DFSMSrmm issues catalog locates as required to check whether data sets are cataloged. *Catalog locates* use the standard catalog search to find out whether a data set is cataloged. Any data sets that are cataloged using JOBCAT or STEPCAT might not be found to be cataloged, because the standard catalog search might not find them.

When you run DFSMSrmm with user catalogs and the DFSMSrmm control data set synchronized, DFSMSrmm does not need to issue catalog locates to find out whether a data set is cataloged. The catalog status tracked by DFSMSrmm in the control data set is used to determine whether a data set is cataloged. When DFSMSrmm tracks catalog status, it does so regardless of whether JOBCAT or STEPCAT is used. During catalog synchronization, DFSMSrmm uses the Catalog Search Interface (CSI) to retrieve data set catalog information. CSI returns catalog information for all data sets in all catalogs that are in or connected to the master catalog. Because of this, DFSMSrmm can detect that a data set is cataloged even if it cannot be found using the standard catalog search.

If you have switched on catalog synchronization, you do not need shared ICF-user catalogs to retain tape data sets as long as they are cataloged. If you have unshared ICF-user catalogs, you have to split your daily inventory management. This is because you have to run the expiration processing on each system that has unshared ICF-user catalogs to return expired volumes back to scratch status. In Figure 2-37, the ICF-user catalogs are shared between systems SYS1 and SYS2, but not with SYS3, so you have to run the expiration processing on either system SYS1 or SYS2, and on system SYS3. All other functions can run on any system sharing the DFSMSrmm control data set.

For more details, see the “Maintaining the Control Data Set” chapter in *DFSMSrmm Implementation and Customization Guide*, SC26-7405, or Chapter 9, “Catalog synchronization” on page 329.

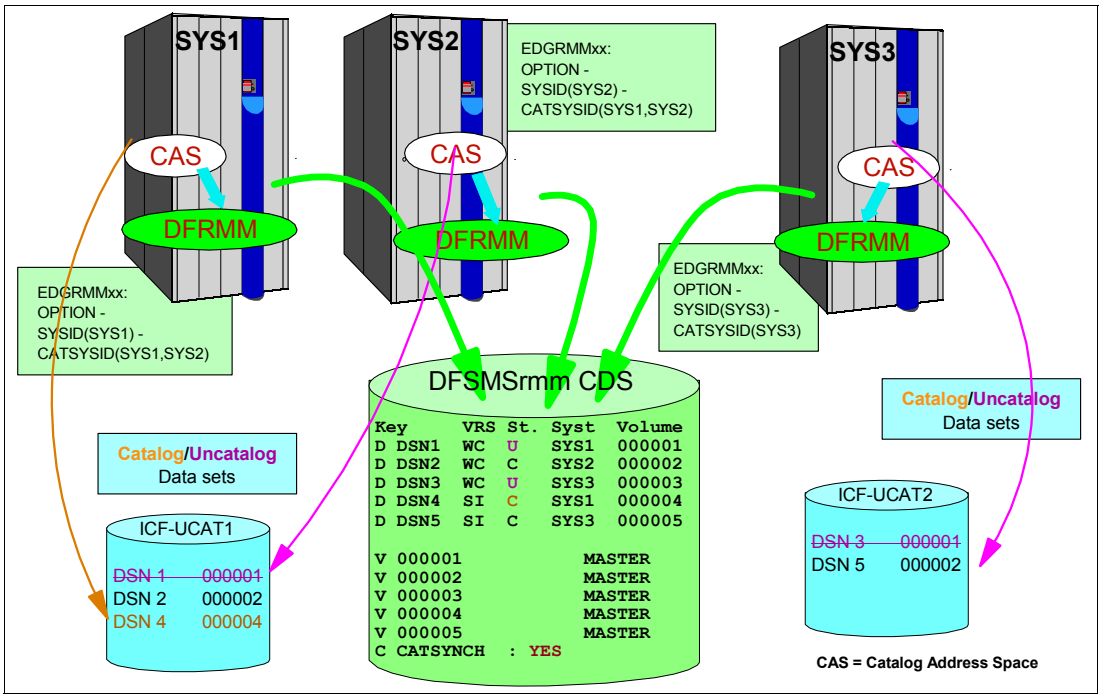


Figure 2-37 Catalog status tracking

2.13 Implementing the RMMplex

With DFSMSrmm z/OS V1.6 or later, you can share a single DFSMSrmm control data set between multiple single systems or multiple sysplexes without needing a shared DASD environment. This new function announced with DFSMSrmm z/OS V1.6 is called *client/server support*.

To use this new function, you must have TCP/IP installed on all systems that share a single DFSMSrmm control data set. Also, all systems must be on z/OS V1.6 or later. For more information, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Restriction: All systems (client and server) must be running with z/OS V1.6 or later. Platforms other than z/OS are not supported.

2.13.1 RMMplex

The RMMplex is one or more MVS systems each running a DFSMSrmm subsystem that shares a control data set. The RMMplex can optionally include one or more DFSMSrmm subsystems as servers, and one or more client subsystems, in addition to standard DFSMSrmm subsystems. The server subsystems and standard subsystems have direct access to and share the DFSMSrmm control data set. The client systems have no direct access to the DFSMSrmm control data set but share the control data set via the server. All systems that share a control data set in this way are part of the same RMMplex.

Figure 2-38 on page 67 shows a complex RMMplex implementation with two sysplexes and one non-plex sharing one DFSMSrmm control data set.

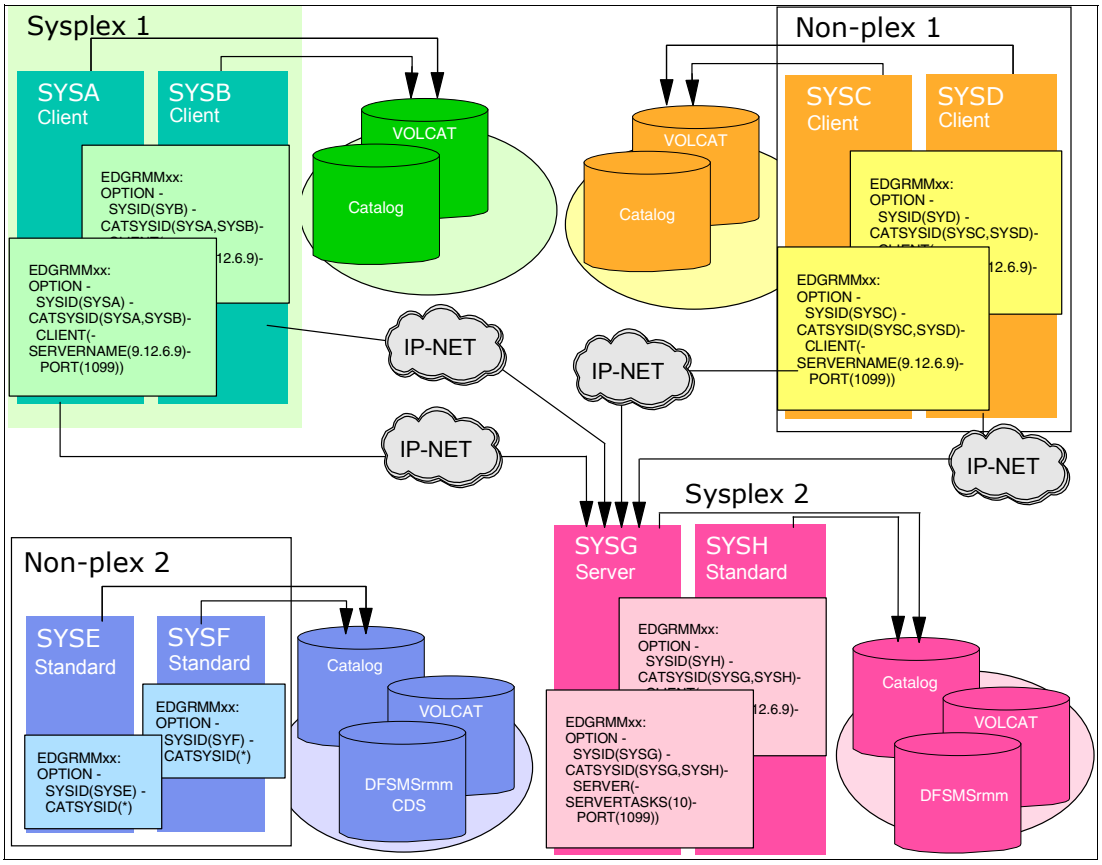


Figure 2-38 RMMplex with six systems sharing one DFSMSrmm CDS

The second non-plex shows a normal shared DFSMSrmm control data set. Each plex has a unique DASD environment that is not shared between the other plexes. Each plex also has its own MVS catalog environment.

DFSMSrmm server

A *server* in DFSMSrmm terminology is an MVS system where DFSMSrmm is up and active. This DFSMSrmm has direct access to the DFSMSrmm control data set and supports requests coming from other systems via TCP/IP.

DFSMSrmm client

A *client* in DFSMSrmm terminology is an MVS system where DFSMSrmm is up and active. This DFSMSrmm has no direct access to the DFSMSrmm control data set and sends all requests to the DFSMSrmm control data set via TCP/IP to the server specified in a EDGRMMxx PARMLIB member.

3

Preparing the environment

This chapter explains how to start Data Facility System Managed Storage removable media manager (DFSMSrmm) on your system and test DFSMSrmm functions without affecting your existing environment.

This chapter includes the following topics:

- Implement and start DFSMSrmm on a test system or in a test environment
- Provide a base for learning about DFSMSrmm

By the end of this chapter, you will have DFSMSrmm started in a test system and have set it up for basic operations in your environment. If DFSMSrmm is running without any errors, you must repeat these steps on your production systems.

3.1 DFSMSrmm implementation

This section guides you through the process of getting DFSMSrmm started on your system with minimal effort and without affecting any existing tape management system that you might have. The intention is to perform only essential customization and setup in order to get DFSMSrmm started quickly.

It is essential that you introduce DFSMSrmm in your installation in a way that avoids any impact to your current processing. It is possible to do that, but you must select the correct DFSMSrmm PARMLIB options and ensure that the correct installation-wide exits are in use. When you start DFSMSrmm, your existing tape management system can continue to run and be unaffected.

Although we advise that you use test data and test volumes initially with DFSMSrmm, you can start DFSMSrmm by using a control data set (CDS) that you create during the extract and conversion steps. If you start DFSMSrmm with a converted CDS, consider the environment within which you plan to run or test. Consider whether the test environment shares any data, catalogs, or volumes with your production environment. Any changes that DFSMSrmm makes as a result of processing can possibly update shared data. Use the correct DFSMSrmm PARMLIB options to prevent updates to RACF, the tape configuration database (TCDB), and catalogs.

Regardless of how you plan to use DFSMSrmm, when you introduce it into any of your systems, it becomes visible to operators and perhaps some users. Those who will come into contact with DFSMSrmm need to be provided with at least basic information about the product and how it affects them.

Follow the steps to implement DFSMSrmm as documented in this Redbooks publication. For more details, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.1.1 z/OS DFSMSrmm Customization Wizard

This wizard is designed to help you set up DFSMSrmm by selecting a standard setup that uses DFSMSrmm default values or a customized setup that you tailor for your installation. It builds the EDGRMMxx parmlib member, and it provides JCL jobs to set up RACF for DFSMSrmm and jobs to allocate and initialize the data sets that are required by DFSMSrmm. Editions of this wizard are available for these releases:

- z/OS V1R12
- z/OS V1R11
- z/OS V1R10
- z/OS V1R9
- z/OS V1R8
- z/OS V1R7

After downloading the required version of the wizard, unpack the compressed archive for local use.

You can find the DFSMSrmm migration wizard on the following Internet website, according to your installed operating system:

http://www-03.ibm.com/systems/z/os/zos/downloads/

These interactive versions of the *DFSMSrmm Implementation and Customization Guide*, SC26-7405, provide information that you can use to plan your implementation of the latest DFSMSrmm functions and features.

3.1.2 Updating the installation-wide exits

The installation-wide exits that you install with DFSMSrmm replace any previously installed exits. The DFSMSrmm supplied programming interface and exits are listed:

- DFSMSrmm installation exits:

  EDGCVRSX Sample to use the installation exit EDG_EXIT100
  EDGUX100 Sample to use the installation exit EDG_EXIT100
  EDGUX200 Sample to use the installation exit EDG_EXIT200
  EDGUX300 Sample to use the installation exit EDG_EXIT300

- DFSMShsm tape volume programming interface:

  EDGTVEXT Programming interface for use from DFSMShsm or Tivoli Storage Manager

- DFSMSdfp MSGDISP exit for updating tape drive displays:

  IGXMSGEX DFSMSdfp MSGDISP exit

- OAM exits:

  CBRUXENT OAM cartridge entry exit
  CBRUXEJC OAM cartridge eject exit
  CBRUXCUA OAM change use attribute exit
  CBRUXVNL OAM volume-not-in-library installation exit

The EDGDFHSM programming interface is used by DFSMShsm and Tivoli Storage Manager to tell DFSMSrmm which tapes are no longer in use by these products. The equivalent interface between DFSMShsm and your tape management system is provided by the ARCTVEXT exit shipped with this product.

Note: DFSMSrmm samples are provided in AEDGSRC1 or SMPSTS.

Before you install DFSMSrmm, you must determine whether any of the DFSMSrmm-supplied exits are used on your system. If they are used, and you still require the functions after conversion or during parallel running to DFSMSrmm, you must merge your existing exits with the DFSMSrmm-supplied exits.

If you share one or more exits between your tape management and DFSMSrmm, you must check whether updates are needed in any of the exits to achieve coexistence when running in parallel.

There are two ways to do this. You can merge your existing exits or use a router exit, which invokes your existing exit and the DFSMSrmm exit. Or, you modify the sample CBRUXEJC, CBRUCENT, and IGXMSGEX, and then this exit will call your existing exit. Figure 3-1 shows the switch you can replace. Samples of the ARCTVEXT, the OAM CBRUXENT, and CBRUXEJC, and the message IGXMSGEX exits are provided with DFSMSrmm in AEDGSRC1 or SMPSTS.

```
CBRUXEJC TITLE 'DFSMSRMM CBRUXEJC SAMPLE USER EXIT'
&API     SETC  'NO'      Replace 'NO' with 'YES' if API call req'd§09A
&ANOEXIT  SETC 'ANOUXEJC'    Replace ANOUXEJC with required name  §07A
&PARALLEL SETC '&SYSPARM'    Replace &SYSPARM with YES if req'd   §07A
```

Figure 3-1 Switch for running sample DFSMSrmm exits in parallel

If you implemented DFSMShsm and Tivoli Storage Manager, do not use the ARCTVEXT router exit. Use the ADMTVEXT instead. If you are using the ARCTVEXT router exit and you set the call of the ARCTVEXT in your DFSMShsm settings, you receive many DFSMSrmm error messages. In this case, DFSMShsm releases each volume twice in DFSMSrmm, because DFSMShsm calls the EDGDFHSM programming interface first and then calls the ARCTVEXT router exit.

Note: Create one front-end exit for each pair of exits if you want both DFSMSrmm and your tape management system to be called and there is no other solution available.

3.1.3 Implementing JES3 USERMODs

DFSMSrmm provides System Modification Program Extended (SMP/E) for z/OS USERMODs in SAMPLIB that you can apply to the standard JES3 user exits and other JES3 modules. Use EDG3UX71, EDG3UX29, and EDG3UX62 to set up DFSMSrmm with JES3. Install the USERMODs to prevent JES3 from validating certain volume mounts, to update JES3 fetch and mount messages, and to enable the use of no label tape volumes with JES3.

In a JES3 system, JES3 validates volume mounts when JES3 is managing tape drives and performing pre-processing setup. JES3 ensures that scratch tapes have reached their expiration date and that volumes are correctly write-enabled or protected. JES3 validation can conflict with DFSMSrmm and RACF processing. For example, you can use “logical write protect” with IBM tape drives to prevent a user from writing to a tape even if the tape is physically write-enabled. Because DFSMSrmm ensures that all scratch tapes are valid and provides features to ignore the expiration dates on volumes, you might want to prevent JES3 from validating certain volume mounts.

For more information about JES3 implementation, see *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.1.4 Updating and validating the PARMLIB members

Update the following SYS1.PARMLIB members:

IEFSSNxx Subsystem definition

IKJTSOxx TSO/E commands and programs

IFAPRDxx Product enablement policy

EDGRMMxx DFSMSrmm options

COMMNDnn (optional) Commands automatically issued at initialization

SMFPRMxx (optional) System Management Facilities (SMF) parameters

GRSRNLxx (optional) Global resource serialization resourcename lists

IFGPSEDI (optional) Enhanced data integrity

DEVSUPxx (optional) Device support options

IGDSMSxx (optional) Storage management subsystem definition

ALLOCnn (optional) Allocation system defaults

AUTORnn (optional) Auto-reply policy specifications

Perform this step once for each MVS image. Details about these members are described in the *MVS Initialization and Tuning Reference,* SA22-7592. For information about using the MVS commands, see the *MVS System Commands Summary,* SA22-7628.

IEFSSNxx: Subsystem definition

There are considerations when defining the DFSMSrmm subsystem name and started procedure name. Review the information provided in this book before updating the IEFSSNxx parmlib member.

You need to make two changes in IEFSSNxx:

- Define a DFSMSrmm subsystem name to MVS.
- Add the name of the subsystem interface initialization program, EDGSSSI, to fully enable DFSMSrmm.

We suggest that you implement the changes in two steps. Define the DFSMSrmm system at this time (see Figure 3-2). You can add the name of the subsystem interface later. If you make only the first change now, you can choose whether to start the DFSMSrmm procedure and what time best suits your particular situation. If you implement both changes at once, DFSMSrmm rejects all tape mounts until the subsystem procedure is active, or you use the EDGRESET utility to disable the interface.

```
SUBSYS SUBNAME(SMS)         /* SMS                              */
       INITRTN(IGDSSIIN)
SUBSYS SUBNAME(JES2)        /* JES2 PRIMARY SUBSYSTEM START     */
       PRIMARY(YES) START(YES)
SUBSYS SUBNAME(DFRM)        /* Name of the DFSMSrmm subsystem   */
SUBSYS SUBNAME(AOPA)       /* NetView                         */
```

Figure 3-2 Defining the DFSMSrmm subsystem name

If you plan to run DFSMSrmm in parallel with any other tape management system, we suggest that you schedule an IPL to set the DFSMSrmm subsystem enabled in IEFSSNxx (see Figure 3-3) before the migration and parallel-running operation.

```
SUBSYS SUBNAME(SMS)         /* SMS                              */
       INITRTN(IGDSSIIN)
SUBSYS SUBNAME(JES2)        /* JES2 PRIMARY SUBSYSTEM START     */
       PRIMARY(YES) START(YES)
SUBSYS SUBNAME(DFRM)        /* Name of the DFSMSrmm subsystem   */
       INITRTN(EDGSSSI)     /*      activate subsystem          */
SUBSYS SUBNAME(AOPA)       /* NetView                         */
```

Figure 3-3 Defining the DFSMSrmm subsystem name with interface enabled

You can use the MVS system command SETSSI to define the DFRM subsystem dynamically. You can issue the SETSSI command from one of the following consoles:

- A console that has master authority
- A console to which an operator with sufficient RACF authority has logged on

To define the DFSMSrmm subsystem, use the MVS system command shown in Figure 3-4 on page 74.

```
SETSSI ADD,SUBNAME=DFRM
```

Figure 3-4 SETSSI ADD command

The following descriptions apply to Figure 3-4:

ADD Specifies that a subsystem is to be dynamically added
SUBNAME Specifies the subsystem name to be dynamically added

You can define the DFSMSrmm subsystem and activate the subsystem interface dynamically using the MVS system commands shown in Figure 3-5.

```
SETSSI ADD,SUBNAME=DFRM,INITRTN=EDGSSSI
SETSSI ACTIVATE,SUBNAME=DFRM
```

Figure 3-5 SETSSI ADD command and ACTIVATE

The following descriptions apply:

ADD Specifies that a subsystem is to be dynamically added
SUBNAME Specifies the subsystem name to be dynamically added
INITRTN Specifies the name of the subsystem initialization routine

Figure 3-6 shows how you can check that you have successfully added the subsystem by using the MVS system command.

```
DISPLAY SSI
```

Figure 3-6 Check the success of adding the subsystem

Example 3-1 shows the output of the command.

Example 3-1 DISPLAY SSI command output

```
IEFJ100I  19.31.54  SSI DISPLAY 121
 SUBSYS=JES2 (PRIMARY)
    DYNAMIC=YES     STATUS=ACTIVE       COMMANDS=REJECT
 SUBSYS=MSTR
    DYNAMIC=NO      STATUS=ACTIVE       COMMANDS=N/A
 SUBSYS=SMS
    DYNAMIC=YES     STATUS=ACTIVE       COMMANDS=REJECT
 SUBSYS=J21A
    DYNAMIC=YES     STATUS=INACTIVE     COMMANDS=REJECT
 SUBSYS=RACF
    DYNAMIC=YES     STATUS=ACTIVE       COMMANDS=REJECT
 SUBSYS=OAM1
    DYNAMIC=YES     STATUS=INACTIVE     COMMANDS=REJECT
 SUBSYS=BP01
    DYNAMIC=YES     STATUS=INACTIVE     COMMANDS=REJECT
 SUBSYS=DFRM
    DYNAMIC=YES     STATUS=ACTIVE       COMMANDS=REJECT
 SUBSYS=OPCC
    DYNAMIC=YES     STATUS=ACTIVE       COMMANDS=REJECT
```

To deactivate the DFSMSrmm subsystem DFSMSrmm interface dynamically, use the MVS system command that is shown in Figure 3-7.

```
SETSSI DEACTIVATE,SUBNAME=DFRM
```

Figure 3-7 MVS command to deactivate the DFRM subsystem interface

IKJTSOxx: TSO/E commands and programs

Update IKJTSOxx to authorize DFSMSrmm commands and utilities. If you use the TSO CSECT facility rather than IKJTSOxx, see *TSO/E Customization,* SA22-7783, for more information.

To authorize the TSO RMM command, add the statement in Figure 3-8.

```
AUTHCMD NAMES(...
              RMM              /* DFSMSrmm TSO commands         */ +
              ... )
```

Figure 3-8 IKJTSOxx operand AUTHCMD

To authorize the DFSMSrmm utilities that you can use within TSO, or call through the TSO Service Facility, add the statements shown in Figure 3-9 and Figure 3-10.

```
AUTHTSF NAMES(...
              EDGAUD           /* DFSMSrmm utility              */ +
              EDGBKUP          /* DFSMSrmm utility   2010/09/07 */ +
              EDGHSKP          /* DFSMSrmm utility              */ +
              EDGRPTD          /* DFSMSrmm utility              */ +
              EDGUPDT          /* DFSMSrmm utility   2010/09/07 */ +
              EDGUTIL          /* DFSMSrmm utility              */ +
              ... )
```

Figure 3-9 IKJTSOxx operand AUTHTSF

```
AUTHPGM NAMES(...
              EDGAUD           /* DFSMSrmm utility              */ +
              EDGBKUP          /* DFSMSrmm utility   2010/09/07 */ +
              EDGHSKP          /* DFSMSrmm utility              */ +
              EDGRPTD          /* DFSMSrmm utility              */ +
              EDGUPDT          /* DFSMSrmm utility   2010/09/07 */ +
              EDGUTIL          /* DFSMSrmm utility              */ +
              ... )
```

Figure 3-10 IKJTSOxx operand AUTHPGM

To use the updated version of the IKJEFTxx member dynamically, use one of the commands in your TSO session or on the MVS console that is shown in Figure 3-11.

```
TSO PARMLIB UPDATE(xx)
SET IKJTSO=xx
```

Figure 3-11 Update IKJEFT dynamically

See Figure 3-12 for an example of how you can check the successful addition of the RMM command and utilities with the TSO or an MVS command.

```
TSO PARMLIB LIST
DISPLAY IKJTSO,ALL
```

Figure 3-12 Display the TSO PARMLIB settings

Example 3-2 shows the output of the TSO PARMLIB LIST command.

Example 3-2 Output of the TSO PARMLIB LIST command

```
TSO/E PARMLIB SETTINGS :

  SYS1.SYSPROG.PARMLIB(IKJTS000) on volume SBOX01
  Activated by **IPL** on 2012-10-13 at 15:51:13 from system SC70
  Applies to :    SC70
...
     CURRENT PARMLIB SETTINGS FOR AUTHCMD:

     ADYOPCMD  ADA       EXITEDIT  RACTRACE  RACTR     RECEIVE
     TRANSMIT  XMIT      LISTB     LISTBC    SE        SEND
     CONSOLE   CONSPROF  PA        PERMALOC  CONCATPA  PARMLIB
     IKJPRMLB  TESTAUTH  TESTA     RMM       MVPXDISP  NETSTAT
     TRACERTE  PING      LIBLIST   Q         QCBXA     QCBESA
     DITTO     DITTOA    DITTOU    TL        RACONVRT  SYNC
     LISTD     LISTDS    RACONVRT  CONSPROF  AD        ADDSD
     AG        ADDGROUP  AU        ADDUSER   ALG       ALTGROUP
     ALD       ALTDSD    ALU       ALTUSER   BLKUPD    CO
     CONNECT   DD        DELDSD    DG        DELGROUP  DU
     DELUSER   LD        LISTDSD   LG        LISTGRP   LISTDATA
     LDATA     LU        LISTUSER  RALT      RALTER    RDEF
     RDEFINE   RDEL      RDELETE   RE        REMOVE    RL
     RLIST     RACLINK   RACDCERT  RACMAP    RVARY     PASSWORD
     PW        PE        PERMIT    SHCDS     SETCACHE  SETR
     SETROPTS  SINFO     SR        SEARCH    BACKUP    OPINFO
     OPSTAT    SRSTAT    WSSTAT    DIAGNOSE  EXAMINE   FREMIGID
     HCEMCS    ARMDREG   CQUERY    CESTPATH  CESTPAIR  CSUSPEND
     CDELPAIR  CDELPATH  CRECOVER  XQUERY    XSTART    XEND
     XSET      XADDPAIR  XDELPAIR  XSUSPEND  XRECOVER  XQML
     SAPITEST  IHSBSNFR  IRRDPI00  CSFTTKE   VLFNOTE   FCESTABL
     FCQUERY   FCWITHDR  MSDPX     LPR

     CURRENT PARMLIB SETTINGS FOR AUTHPGM:

     ARY$TSOC  AMASPZAP  AOPBATCH  CC        EXITEDIT  HESEVMI
     HESEVMX   ICKDSF    IEBCOPY   IEBGENER  IFCEREP1  Q
     QCBXA     QCBESA    DRMAUTH   EDGAUD    EDGBKUP   EDGHSKP
     EDGRPTD   EDGUPDT   EDGUTIL   FEC$TSOC  HAA$EXCP  ICHDSM00
     ICHUT100  ICHUT200  ICHUT400  IDCAMS    IRRDSC00  IRRIRA00
     IRRUT100  IRRUT200  IRRUT400  ITPENTER  CSFDAUTH  CSFDPKDS
     IXCMIAPU  SINFO     DITTOA    IGWPIT    IEARELCN  ARMDREG
     BDGJV1    SAPITEST  IRRDPI00  IOEGRWAG  IOENEWAG  IOESALVG
     IXCMIAPU  IXCM2APU  ADB2ATH   ADB2UTIL

       CURRENT PARMLIB SETTINGS FOR AUTHTSF:
```

```
    ARY$TSOC  IEBCOPY   ICQASLIO  ICQASLCO  DRMAUTH   IHVUUSD
    ITPENTER  CSFDAUTH  CSFDPKDS  EQQMINOR  EQQMINOH  EDGAUD
    EDGBKUP   EDGHSKP   EDGRPTD   EDGUPDT   EDGUTIL   FEC$TSOC
    IAMRECVR  IAMXMONA  FMNSMF    HAA$EXCP  ADB2ATH   ADB2UTIL
    CAT01IKB  IKJEFF76
  ....
```

IFAPRDxx: Product enablement policy

When an installation order requires an IBM product, such as z/OS, that provides product enablement, IBM supplies a tailored IFAPRD00 member of SYS1.PARMLIB. This tailored member enables the product and any optional features ordered with the product. IFAPRDxx is the parmlib member that ensures that you are licensed to use the DFSMS components you want. IFAPRDxx replaced IGDDFPKG in OS/390 Version 1, Release 1.

Review your IFAPRDxx parmlib member to verify that IFAPRDxx was updated during installation to reflect that DFSMSrmm is licensed for your use (see Figure 3-13). For more information about IFAPRDxx, see *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

```
PRODUCT OWNER('IBM CORP')
        NAME(OS/390)
        ID(5647-A01)
        VERSION(*) RELEASE(*) MOD(*)
        FEATURENAME(DFSMSRMM)
        STATE(ENABLED)
```

Figure 3-13 Enabling DFSMSrmm license into IFAPRDxx

You can also define a global product enablement policy setting for all products, as shown in Figure 3-14.

```
WHEN (SYSNAME(*))
     PRODUCT NAME(*) STATE(ENABLED)
```

Figure 3-14 Global product enablement policy

You can use the MVS SET PROD operator command (see Figure 3-15) to modify the enablement policy dynamically by specifying which IFAPRDxx member the system is to use. Statements in the IFAPRDxx member modify an existing policy instead of replacing it. The change to the policy takes place immediately but does not affect any product instances that are already running. To update the IFAPRDxx member dynamically, use the following MVS system command.

```
SET PROD=xx
```

Figure 3-15 SET PROD command

Figure 3-16 on page 78 shows how to use the MVS DISPLAY command to display the active enablement policy.

```
DISPLAY PROD,REGISTERED
or
DISPLAY PROD,STATE
```

Figure 3-16 DISPLAY PROD command

Example 3-3 shows the output of a DISPLAY PROD command.

Example 3-3 Output of a DISPLAY PROD command

```
RESPONSE=SC70
 IFA111I 19.53.51 PROD DISPLAY 033
 S OWNER            NAME             FEATURE          VERSION  ID
 E IBM CORP         z/OS             z/OS             01.12.00 5694-A01
 E IBM CORP         z/OS             DFSMSHSM         **.**.** 5694-A01
 E IBM CORP         z/OS             DFSMSRMM         **.**.** 5694-A01
 E IBM CORP         z/OS             DFSMSTVS         **.**.** 5694-A01
 N IBM CORP         z/OS             JES2             01.12.00 5694-A01
 N IBM CORP         z/OS             RACF             **.**.** 5694-A01
 E IBM CORP         z/OS             RMF              **.**.** 5694-A01
 E IBM CORP         z/OS             Security Server  **.**.** 5694-A01
 E IBM CORP         z/OS             SDSF             **.**.** 5694-A01
 E IBM CORP         z/OS             TCP/IP BASE      **.**.** 5694-A01
```

Example 3-4 shows the result if you have a global product enablement policy setting.

Example 3-4 Output of a DISPLAY PROD command with global product enablement policy

```
RESPONSE=SC70
 IFA111I 18.24.13 PROD DISPLAY 639
 S OWNER            NAME             FEATURE          VERSION  ID
 E *                *                *                * .* .*  *
```

IFGPSEDI: Enhanced data integrity

The purpose of enhanced data integrity in the IFGPSEDI member of SYS1.PARMLIB is to detect programs that cause data loss due to multiple programs updating a sequential data set simultaneously. This is a useful function. However, if your system programmer chooses to activate it, DFSMSrmm receives serious warning messages or ABENDs.

To activate the enhanced data integrity function, the system programmer performs these actions:

1. Create SYS1.PARMLIB member IFGPSEDI with MODE(WARN) or MODE(ENFORCE).
2. Re-IPL the system or enter the command shown in Figure 3-17.

```
S IFGEDI
```

Figure 3-17 Set the enhanced data integrity function

You can specify one of three different mode options:

DISABLE If the member has MODE(DISABLE) or MODE is omitted, the member has no effect.

WARN If MODE(WARN) is in effect, during the normal running of DFSMSrmm programs, the system issues message IEC984I or message IEC985I

for the DFSMSrmm journal, ACTIVITY, MESSAGE, REPORT, REPTEXT, and XREPTEXT data sets. You can ignore these messages. To suppress these warning messages, code those names in the DSN parameter in the IFGPSEDI member.

ENFORCE If MODE(ENFORCE) is in effect, normal running of DFSMSrmm programs causes ABEND 213-FD for the DFSMSrmm journal, ACTIVITY, MESSAGE, REPORT, REPTEXT, and XREPTEXT data sets. You must code the names of these data sets in the DSN parameter in the IFGPSEDI member to suppress these OPEN failures. Alternately, you can change MODE(ENFORCE).

EDGRMMxx: DFSMSrmm options

You need to create an EDGRMMxx PARMLIB member for DFSMSrmm (see Example 3-5 on page 96). The member name is in the form EDGRMM*xx*, where *xx* is a two-character alphanumeric suffix of your choice. The default is EDGRMM00. A sample EDGRMMxx member is provided in SAMPLIB member EDGIVPPM. You can copy it to PARMLIB and modify it according to your installation requirements.

In the following sections, we describe the most important options to specify in the EDGRMMxx PARMLIB member when DFSMSrmm is running in parallel to your tape management system. Use the following information together with the *DFSMSrmm Implementation and Customization Guide*, SC26-7405, which describes in detail the options you can use with DFSMSrmm.

OPTION command

This section details the various OPTION commands that are used within EDGRMMxx. We suggest that you specify the following OPTION subparameters in EDGRMMxx:

- **OPMODE(R)**

  Specifies the mode of DFSMSrmm:

  R / Record-only mode

  You can use DFSMSrmm TSO commands, ISPF dialog, and inventory management functions. In addition, DFSMSrmm records information about magnetic tape volumes used on the system, including details about volume owners and data set names.

  For volumes in an Automated Tape Library Dataserver, DFSMSrmm records information about entry and eject activities. During entry processing, DFSMSrmm provides volume status information to the object access method (OAM). If a volume is not already defined to DFSMSrmm, DFSMSrmm automatically defines it as long as there are no conflicts with either the VOLSER or corresponding rack number.

  DFSMSrmm does not validate or reject volumes when in record-only mode. When DFSMSrmm is running in record-only mode, the volume status in the TCDB is not updated. When you issue DFSMSrmm commands, the TCDB is updated if appropriate and the volume is system-managed. When volumes are returned to scratch status, no TCDB updates are made.

  Table 3-1 on page 80 and Table 3-2 on page 80 provide information about the options that are affected by the OPMODE value.

Table 3-1 How the OPMODE value honors the settings of various options

Option	Manual	Record only	Warning	Protect
EXPDTCHECK	N	N	Y	Y
UNCATALOG	Y	Y	Y	Y
TPRACF	Y	Y	Y	Y
SMSTAPE(UPDATE)	Y	Y	Y	N
SMSTAPE(PURGE)	N	Y	Y	Y

Table 3-2 How the OPMODE value affects system-managed tape library support

Main area of activity	Manual	Record only	Warning	Protect
Command processing	OPT	OPT	OPT	YES
Expiration processing	OPT	OPT	OPT	YES
Support for CBRUXxxx exits	N/A	OPT	OPT	YES
Purging TCDB records during eject processing	N/A	OPT	OPT	YES
Legend: ► N/A: function not available. ► OPT: Function can be controlled by the SMSTAPE option. ► YES: Function is always provided.				

- **BACKUPPROC(EDGBKUP)**

  Specifies the name of the procedure that you want to start automatically when the journal percentage full threshold is reached. Specify a valid alphanumeric procedure name 1 - 8 characters. If no name is specified, no automatic start command is issued.

- **CDSID(PROD)**

  Specifies the identifier of the control data set that must be used on this system. Specify a value 1 - 8 characters long. This ID is used by DFSMSrmm web services to distinguish retrieved data between multiple control data sets. Ensure that each DFSMSrmm control data set has a unique CDSID value.

- **COMMANDAUTH(OWNER)**

  Specifies the type of authorization that DFSMSrmm is to check. Specify OWNER when you expect the owners of volume information and data set information to be able to update their own data sets and volumes using RMM TSO subcommands. Specify DSN when you expect changes to volume and data set information to be authorized with the RACF DATASET class and TAPEVOL class.

- **DATEFORM(J)**

  Specifies the date format for messages and reports. Use the date format preferred in your computing center, as shown in Table 3-3 on page 81.

Table 3-3 DATEFORM options

Value	Language	Format	Example
A	American	*mm/dd/yyyy*	12/15/2010
E	European	*dd/mm/yyyy*	15/12/2010
I	ISO	*yyyy/mm/dd*	2010/12/15
J	Julian	*yyyy/ddd*	2010/349

- **DISPDDNAME(LOANDD)**

  Specifies the name of the DD card that identifies the data set, which contains disposition control statements that DFSMSrmm processes during CLOSE or end-of-volume (EOV) processing. For detailed information, see 7.6, “Disposition processing” on page 250.

- **DISPMSGID(EDG4054I)**

  Specifies the message number that DFSMSrmm uses for write-to-operator messages that are specified in the disposition control file.

- **DSNAME(RMM.PROD.CDS)**

  Specifies the name of the DFSMSrmm CDS. Specify a name up to 44 characters long.

  Note: If you do not specify DSNAME, you must specify the data set name in the MASTER DD statement in the DFSMSrmm started procedure. If you specify a name for both DSNAME and MASTER DD, DFSMSrmm ignores the MASTER DD statement.

- **EXPDTDROP(PERCENT(10,WARN))**

  Use EXPDTDROP to specify a maximum number or percentage of existing expiration date retained volumes that can be dropped from retention and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of EXPDT-retained physical and logical volumes at the start of inventory management expiration processing and the number of these to be set to pending release. An EXPDT-retained volume is one that is not vital record specification (VRS) retained and is not newly assigned. When you specify count, this is an absolute maximum number of volumes that can be released by a single run of EDGHSKP EXPROC processing. When you specify a percentage, this is a maximum percentage of the existing EXPDT-retained volumes that can be released by a single run of EDGHSKP EXPROC processing. This processing occurs each time that you run inventory management EXPROC processing.

  When VRSEL and EXPROC are run together in a single EDGHSKP run, volumes that are dropped by VRSEL processing are counted only towards the VRSDROP limit, not to the EXPDTDROP limit.

- **GDG(CYCLEBY(GENERATION) DUPLICATE(BUMP))**

  Use the GDG option to specify how generation data groups (GDG) are handled for cycle retention by VRSEL processing. Cycle retention includes both the CYCLES and the BYDAYSCYCLE retention types. The correct sequence for determining the retention can be either by using the generation number or the creation order. You can also specify how duplicate generations (generation data sets) are handled and have the flexibility to include or exclude duplicate generations from the cycles count as required by your application processing.

The following values apply:

CYCLEBY(GENERATION)
Specifies that retention is to be based on the generation number. DFSMSrmm determines the generation number by applying a similar algorithm to that used by catalog processing. Both the creation order and the generation number from the data set name are considered, allowing wraps in generation number to be correctly handled.

CYCLEBY(GENERATION) is the default.

DUPLICATE(BUMP) Specifies that a duplicate generation is to be bumped by the current VRS subchain and considered for retention by a subsequent VRS subchain.

DUPLICATE(BUMP) is the default.

- **JRNLNAME(RMM.PROD.JRNL)**

  Specifies the name of the journal. Specify a name up to 44 characters long.

  Note: If you do not specify JRNLNAME, you can specify a name in the JOURNAL DD statement in the DFSMSrmm started procedure. If you do not specify a journal name in either EDGRMMxx or the started procedure, DFSMSrmm does not provide journaling. If you specify a name in both, DFSMSrmm uses the JRNLNAME value and ignores the JOURNAL DD statement.

- **JRNLTRAN(NO)**

  Use the JRNLTRAN operand to specify whether the unchanged copy of a CDS record is journaled, as well as the updated copy.

  The following values apply:

  NO Specifies that only the updated copy of the CDS record is to be journaled.

  YES Specifies that the pre-update copy of the CDS record being updated is to be journaled, in addition to the updated copy of the CDS record. Set this option only on a test or recovery system when you plan to use the EDGUPDT utility to duplicate CDS record updates back in the production CDS. As a result of using this option, you need to plan on providing up to 33% more journal data set space to accommodate the additional records.

- **MASTEROVERWRITE(LAST)**

  Specify to control how DFSMSrmm allows the overwriting of a volume.

  The following values apply:

  ADD Specify this value so new data can be created and no existing data can be destroyed. No existing file on a volume can be re-created, but the last file can have new data added to it. When adding data to the last file, DFSMSrmm checks that the data set name used must match the existing data set name. Select this option when you want the last file on the volume to be extended or a new file added to the volume.

Note: DFSMSrmm enforces the MASTEROVERWRITE(ADD) option on a WORM tape that is in master status. This is done to ensure that you see a message from DFSMSrmm rather than one of a number of symptoms as a result of the tape drive preventing overwrites.

LAST Specify this value to ensure that when an existing file on a master volume is being written to that only the last file on the volume can be used. The data set name used must match the existing data set name. Select this option when you want the last file on the volume to be used for output.

MATCH Specify this value to ensure that when an existing file on a master volume is being used for output that exactly the same data set name must be used. Select this option when you want any existing file on the volume to be re-created regardless of whether it is the last file on the volume as long as the same data set name is used. When you use an existing tape file for output, all the files that are higher in sequence are destroyed.

USER Specify this value to allow any existing file on a master volume to be used for output regardless of the data set names being used and the existing tape file's relative file position on the volume. Select this option when you want validation of master volumes to be just the same as for user status volumes. When you use an existing tape file for output, all the files that are higher in sequence are destroyed.

The default value is MASTEROVERWRITE(LAST).

- **MAXHOLD(100)**

  Specifies the maximum number of activities DFSMSrmm performs to the CDS before the reserve is released and reacquired. Specify MAXHOLD to minimize the impact that long operations, such as RMM TSO ADD subcommands with large count values, or large searches, have on other users.

- **MAXRETPD(NOLIMIT)**

  Specifies the maximum retention period that a user can request for data sets on volumes. Specify NOLIMIT or a value between 0 and 93000 days. When a value between 0 and 93000 days is specified, the value is added to the current date to determine the maximum allowed expiration date.

  The following values apply:

  NOLIMIT Specify NOLIMIT to use the dates 99365 or 99366, which means the data sets never expire. If the calculated date is 31 December 1999, the expiration date 1 January 2000 is used.

  If the DFSMSrmm ISPF dialog or RMM TSO subcommands are used to specify a retention period or expiration date that exceeds the MAXRETPD value, DFSMSrmm fails the subcommand or panel request.

- **MEDIANAME(3490)**

  Specify MEDIANAME to set a default *medianame* value that DFSMSrmm uses when you do not specify a media name for a volume. The media name is used when you add volumes, define pools in your installation, define a default pool for your installation, and when the EDGINERS utility selects volumes for automatic processing.

- **MOVEBY(VOLUME)**

  Specify the MOVEBY operand to move volumes that are retained by DFSMSrmm VRSs as a set of volumes or as individual volumes.

  The following values apply:

  SET Specify this value when you want volumes moved as a set. DFSMSrmm moves all volumes in the same multivolume set that are retained by VRSs. All the volumes are retained in the same location selected by the VRS priority or location priority for the volume.

  VOLUME Specify this value when you want volumes moved as individual volumes. DFSMSrmm moves the volumes without considering the location of the other volumes in the multivolume set.

  PREACS(NO) Specify the PREACS operand to control whether DFSMSrmm-supplied and EDGUX100 installation exit-supplied values are input to storage management subsystem (SMS) pre-automatic class selection (ACS) processing.

  The following value applies:

  NO Specify NO to avoid DFSMSrmm pre-ACS processing using the DFSMSrmm EDGUX100 installation exit.

- **RETAINBY(VOLUME)**

  Use the RETAINBY option for volumes that are managed by the VRSEL retention method to specify whether DFSMSrmm retains multivolume sets as a set or as individual volumes.

  Note: For volumes managed by the EXPDT retention method, use the RETAINBY suboption of the RETENTIONMETHOD(EXPDT) option to obtain a similar function.

  The following values apply:

  SET When you retain by set, if any volume in a set is retained by a VRS, all volumes in the set are retained as vital records. DFSMSrmm uses the highest retention date of all volumes in the set as the retention date for all volumes that are retained as vital records in a set. If no volume in a set is retained by a VRS, DFSMSrmm performs expiration processing by set. DFSMSrmm does not expire volumes in a set if at least one volume in a set is still not ready to expire because it has not reached its expiration date and you have not specified that you want the expiration date ignored.

  VOLUME When you retain by volume, DFSMSrmm retains a volume based on VRSs and on the volume expiration date. DFSMSrmm does not consider other volumes in the set.

- **RETENTIONMETHOD(VRSEL)**

  Use this operand to set the system-wide retention method for new tape volume sets created during Open/Close/End of Volume (O/C/EOV) processing, and for tape volumes added to the DFSMSrmm CDS. RETENTIONMETHOD can be abbreviated as RM.

  The following value applies:

  VRSEL Specify VRSEL to set the default retention method for new tape volume sets to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to VRSs, and if a match is found, to determine whether the data set

or volume is to be retained by VRS. The VRSEL retention method is controlled by all the other VRS-related options in parmlib, including OPTION RETAINBY and MOVEBY.

- **RETPD(5)**

  Specifies the default retention period for all new data sets on volumes. Specify a value between 0 and 93000 days. The specified value is added to the current date to determine the expiration date. Select a default retention for parmlib RETPD that is a small value to ensure that all tape data created outside the service levels is released as soon as possible. The MAXRETPD value you specify in the parmlib limits the calculated expiration date.

  Note: The value that you specify for the RETPD operand needs to be the same as the value you use in your tape management system.

- **RETENTIONMETHOD(VRSEL)**

  Use this operand to set the system-wide retention method default for new tape volume sets. New tape volume sets can be created during Open/Close/End of Volume (O/C/EOV) processing, or through DFSMSrmm commands. A tape volume set can be a multivolume set, or a single tape volume.

  The following values apply:

  EXPDT
  Specify EXPDT to set the default retention method for new tape volume sets to be based on EXPDT. Data sets and volumes that are managed by this retention method are never processed by VRSEL inventory management. When you specify the EXPDT retention method the DFSMSrmm inventory management EXPROC processing always attempts to return volumes to scratch on the same run as the volume is released (this is as if the SCRATCHIMMEDIATE attribute is set for the volume). DFSMSrmm maintains a consistent expiration date and time for all data set records of a multivolume data set, unless the volume set is retained by first file.
  EXPDT can be specified either as EXPDT or EXPDT(*options*).

  The available options are listed:

  LASTREF(*extra_days*)
  LASTREF specifies the default for the data set record LASTREF attribute. The LASTREF attribute specifies the number of days that the data set will be retained after the data set was last referenced by a read or write operation. LASTREF applies only to data sets on volumes that are managed by the EXPDT retention method.

  extra_days
  A decimal number between 0 and 93000. The value must not exceed the maximum retention period (MAXRETPD) specified in the EDGRMMxx parmlib member. An *extra_days* value of 0 has the same effect as the NOLASTREF operand.

  When a data set is added to DFSMSrmm on a volume that is managed by the EXPDT retention method and neither LASTREF or NOLASTREF is specified for the data set, DFSMSrmm uses the default LASTREF value.

DFSMSrmm uses the data set LASTREF value to determine the data set expiration date. The extra days are added to the date of last reference. The data set expiration date is set to the maximum of the date that is calculated using the data set *LASTREF* value and the date resulting from applying the EXPDT, RETPD, or default RETPD. Any reference to the data set by a read or write operation will change the expiration date.

If neither LASTREF or NOLASTREF is specified in parmlib, NOLASTREF is used by default.

NOLASTREF
NOLASTREF is the default setting for the data set record LASTREF attribute. NOLASTREF specifies that DFSMSrmm does not consider the data set last reference date when determining the data set expiration date. NOLASTREF applies only to data sets on volumes that are managed by the EXPDT retention method.

RETAINBY(FIRSTFILE | SET | VOLUME)
RETAINBY specifies how DFSMSrmm is to retain volumes or multivolume sets that are managed by the EXPDT retention method:

FIRSTFILE
The expiration date of the first file is used to set the expiration date of the volume or multivolume set. All volumes in a set will have the exact same expiration date and will be released to scratch in the same run of DFSMSrmm inventory management.

SET
The expiration date is the maximum of all data set expiration dates of all the volumes in the set. All volumes in the set will have the exact same expiration date. Any file on any volume of the set can increment the volume expiration date. All files of a multivolume data set have the same expiration date.

VOLUME
The expiration date is determined separately for each volume in the set. Unless defined differently, the expiration date is the maximum of all data set expiration dates on the volume. Each file on a volume can increment the volume expiration date. All files of a multivolume data set have the same expiration date.

Note: In a multivolume set, RETAINBY is assigned only to the first volume in a multivolume sequence. All other volumes added to the set assume the same RETAINBY option.

For volumes that are managed by the VRSEL retention method, use the RETAINBY option to obtain a similar function.

VRSEL
Specify VRSEL to set the default retention method for new tape volume sets to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to VRSs, and if a match is found, to determine whether the data set or volume is to be retained by VRS. The VRSEL retention method is controlled by all the other VRS-related options in parmlib, including OPTION RETAINBY and MOVEBY.

- **RETPD(5)**

  Specifies the default retention period for all new data sets on volumes. Specify a value between 0 and 93000 days. The specified value is added to the current date to determine the expiration date. Select a default retention for parmlib RETPD that is a small value to ensure that all tape data created outside the service levels is released as soon as possible. The MAXRETPD value that you specify in the parmlib limits the calculated expiration date.

  DFSMSrmm sets a default retention period in the following manner:

  – If you specify RETPD or EXPDT, the value is used as the data set's new expiration date.

  – If you do not specify RETPD or EXPDT, DFSMSrmm uses the EXPDT or RETPD allocation attribute of a data class, if all these conditions are true:

    - The data set is associated with a data class, either explicitly by the DATACLAS keyword on the JCL or implicitly by an automatic class selection routine.
    - The data class has an EXPDT or RETPD allocation attribute.
    - The storage management subsystem is active.

    DFSMSrmm uses the management class Expire after Date/Days as the data set's new expiration date, if all these conditions are true:

    - You do not specify RETPD or EXPDT in the JCL or data class.
    - The data set is assigned to a management class.
    - The management class Expire after Date/Days is set.
    - The storage management subsystem is active.

    DFSMSrmm uses the default retention period that is set in EDGRMMxx, if RETPD or EXPDT are not otherwise specified.

  Whenever a new data set is written to tape, DFSMSrmm checks whether the volume's expiration date needs to be updated, based on whether the new data set has a longer expiration date than the volume on which it is written. DFSMSrmm gets the expiration date for a data set from the job file control block (JFCB) or from the management class at open time. If there is a date in the JFCB or management class, DFSMSrmm compares this date to the current expiration date for the volume. If the date in the JFCB allows the volume to be retained longer, DFSMSrmm uses that date to update the volume's expiration date.

  If there is no expiration date in the JFCB or management class, DFSMSrmm uses the EDGRMMxx RETPD value to calculate the new expiration date. If the RETPD value allows the volume to be retained longer, DFSMSrmm uses that date to update the volume's expiration date.

  You can set the date in the JFCB in several ways:

  – RETPD and EXPDT keywords in the JCL

  – Data class when the storage management subsystem is active and the volume is system-managed

  – A user program, using the RDJFCB macro and OPEN TYPE=J after modifying the JFCB

  – Installation exits in use on your particular system

  When you use the VRSEL retention method, you can also use VRSs to define more specific default retention periods for users by using a data set name prefix.

  The default is RETPD(5).

- **REUSEBIN(STARTMOVE)**

  Specify to control how DFSMSrmm reuses bins when a volume is moving from a bin. The following values apply:

  STARTMOVE A bin can be reused as soon as a volume starts moving out of a bin. Extended bin support must be enabled before you can use this operand. For more information, see the *DFSMSrmm Implementation and Customization Guide*, SG26-7405, and the chapter in that book titled “Enabling Extended Bin Support” to enable extended bin support.

  Note: The use of STARTMOVE is advised because most vendor tape management systems reuse bins directly.

- **SMFAUD(YES)**

  Specifies the SMF record type to be used for audit records. Specify YES or a number between 128 and 255 that is different from the value for SMFSEC. The value must conform to standard SMF conventions.

  The following value applies:

  YES It is advised that you do not use an SMF record type number *nnn*, but instead use the IBM assigned record number by specifying YES. The IBM assigned record number is type 42, and the subtype is 22. You cannot mix SMF record types. For example, you cannot specify YES for SMFAUD and a record type for SMFSEC.

- **SMFSEC(YES)**

  Specifies the SMF record type to be used for security records. Specify YES or a number between 128 and 255 that is different from the value for SMFAUD. The value must conform to standard SMF conventions.

  The following value applies:

  YES It is advised that you do not use an SMF record type number *nnn*, but instead use the IBM assigned record number by specifying YES. The IBM assigned record number is type 42, and the subtype is 23. You cannot mix SMF record types. For example, you cannot specify YES for SMFAUD and a record type for SMFSEC.

- **SMSACS(YES)**

  Specify this operand to control whether DFSMSrmm calls SMS ACS processing to enable the use of storage group and management class values with DFSMSrmm.

  The following value applies:

  YES Specify YES to enable DFSMSrmm calls to the SMS ACS processing to obtain management class and storage group names. If values are returned by the SMS ACS routines, the values are used instead of the DFSMSrmm and EDGUX100 decisions.

- **SMSTAPE(UPDATE(EXITS, SCRATCH, COMMAND), PURGE(YES))**

  Use SMSTAPE to specify how DFSMSrmm updates the TCDB and controls system-managed tape processing.

The following values apply:

UPDATE Use UPDATE to select the system-managed tape functions DFSMSrmm provides. The UPDATE operand has three subparameters: EXITS, SCRATCH, and COMMAND. You can specify one or more of the subparameters. When DFSMSrmm is running in PROTECT mode, DFSMSrmm ignores the UPDATE operand and performs processing as though you specified EXITS, SCRATCH, and COMMAND. When DFSMSrmm is running in WARNING or RECORD mode, DFSMSrmm does not update TCDB information unless you request the update. You can specify one or more of the values. When you specify a value, DFSMSrmm performs the updates to the TCDB:

EXITS Specify when you want DFSMSrmm volume status information to override the OAM volume status during entry processing and you want to use the DFSMSrmm VNL exit.

SCRATCH Specify when you want DFSMSrmm to update the volume status in the TCDB during expiration processing when volumes are returned to scratch status.

COMMAND Specify when you want to use the RMM TSO subcommands or the DFSMSrmm API to update the TCDB. This controls change of status, Tape Device Selection Information (TDSI), and owner information, eject processing, and manual cartridge entry processing.

PURGE Use PURGE to control how DFSMSrmm affects the TCDB volume records during EJECT processing. The default is PURGE(ASIS) in all operating modes except MANUAL mode. In manual mode, DFSMSrmm provides no support for eject processing.

YES Specify when you want DFSMSrmm to force the TCDB volume records to be purged at eject time.

- **TPRACF(N)**

  Do not create RACF tape profiles. See Chapter 13, “System Authorization Facility tape security” on page 475.

  Important: DFSMSrmm manipulates your RACF TAPEVOL and TAPEDSN profiles in each operating mode if the TPRACF function is set to A or P.

- **TVEXITPURGE**

  Specifies how DFSMSrmm processes volumes to be purged by callers of EDGTVEXT or EDGDFHSM.

  The following values apply:

EXPIRE(*days*) Use the EXPIRE(*days*) option to set the volume expiration date to the current date plus days for volumes to be purged. Use the EXPIRE(*days*) parameter when you use the EXPDT retention method, using days as a way to delay expiration of the volume. When using the VRSEL retention method, you can optionally use this operand in combination with VRSs that use the UNTILEXPIRED retention type. First, the EXPIRE(*days*) is applied to set a new volume EXPDT. Next, by running VRSEL, you can then optionally extend retention using the extra days retention type. For example, specify EXPIRE(0) when using a VRS with extra days

retention or you can use a nonzero EXPIRE(*days*) value and avoid using an extra days retention VRS.

days
: The number of days for which DFSMSrmm retains the volume before considering it for release. The value is a one-digit to four-digit decimal number and is added to today's date to compute the new expiration date. If the value exceeds the maximum retention period (MAXRETPD), it is reduced to the MAXRETPD value.

  The default value for days is 0. That is, TVEXTPURGE(EXPIRE) is the same as TVEXTPURGE(EXPIRE(0)).

NONE
: DFSMSrmm take no action for volumes to be purged.

RELEASE
: DFSMSrmm releases the volume to be purged according to the release actions that are set for the volume. You must run expiration processing to return a volume to scratch status.

  This is the default.

- **UNCATALOG(N)**

  Do not use DFSMSrmm uncatalog processing if DFSMSrmm is not your primary tape management system. If UNCATALOG(N) is set, DFSMSrmm does not uncatalog data sets under any circumstances.

  Important: DFSMSrmm uncatalogs data sets in each operating mode if the UNCATALOG function is set to Y or S.

- **VRSDROP(PERCENT(10,WARN))**

  VRS-retained volumes that can be dropped from vital records retention and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of VRS-retained physical and logical volumes at the start of vital record selection processing and the number of these dropped by vital record processing. When you specify COUNT, this is an absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing. When you specify PERCENT, this is the maximum percentage of the existing VRS-retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

  VRSDROP processing is intended to provide limited checking for volumes that are already VRS-retained. It considers how many of these existing VRS-retained volumes are removed from VRS retention. Those volumes dropped from VRS retention can be set pending release, depending on VRS release options and the volume expiration date. They can also become EXPDT-retained and on the next run of EXPROC, they will be considered by EXPDTDROP limit processing.

- **VRSJOBNAME(2)**

  Specify this operand to select whether DFSMSrmm uses the job name from VRS as the primary or secondary match.

  The following value applies:

  VRSJOBNAME(2)
  : The data set name takes precedence over the job name during VRS match processing. This is the option that most closely matches the use of job names in your tape management system.

- **VRSRETAIN(PERCENT(80,WARN))**

  Use VRSRETAIN to specify a minimum number or percentage of newly assigned volumes that are to be retained by vital records retention and the action to be taken by DFSMSrmm. A newly assigned volume is one that has a volume assignment date and time that is higher than the run date and time of the previous VRSEL processing and that is not VRS-retained. DFSMSrmm counts the number of newly assigned physical and logical volumes at the start of vital record selection processing and the number of these that become VRS retained by vital record processing. When you specify count, this is an absolute minimum number of volumes that are to be retained by a single run of EDGHSKP VRSEL processing. When you specify a percentage, this is the minimum percentage of the newly assigned volumes that are to be retained by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

  VRSRETAIN processing is intended to provide limited checking for volumes that contain newly created data sets that have not yet been processed by inventory management VRSEL. The data sets on the volumes have been created since the start of the last completed VRSEL run. It considers how many of these volumes become VRS-retained during the new VRSEL run. Those volumes that become VRS-retained will be considered by the VRSDROP limit checking in future VRSEL runs. Those volumes that are not retained by VRS can be set pending release, depending on VRS release options and the volume expiration date. They can also become EXPDT retained and on the next run of EXPROC, they will be considered by EXPDTDROP limit processing.

- **VRSEL(NEW)**

  Use VRSEL to control how DFSMSrmm inventory management vital record processing uses retention and movement information defined in VRSs.

  The following value applies:

  NEW

  Specify which function you want:

  - DFSMSrmm to process retention information in NAME VRSs, release options defined in VRSs, and VRSs chained using the RMM ADDVRS operand
  - To convert using the conversion tools and require functions close to functions that are provided by your tape management system

  Note: The VRSEL option OLD is no longer supported.

REJECT command

Important: The new command to control the use of tape volumes is OPENRULE and the new command for partitioning your tape libraries is PRTITION. When you use either the PRTITION or OPENRULE command, the REJECT commands are no longer used. You must start using both PRTITION and OPENRULE at the same time to avoid loss of function.

DFSMSrmm can reject a range of tapes only if you first define them to DFSMSrmm. It can reject all tapes not defined to it, which you specify with a single asterisk for the prefix, but it cannot reject a range of tapes not defined to it.

You can specify multiple REJECT commands to define different ranges of tapes. However, use only one ANYUSE operand and one OUTPUT operand for each REJECT command.

The following value applies:

ANYUSE(*) Specifies the shelf locations of volumes not available for read or write processing. You can specify a generic shelf location consisting of a one to five character prefix followed by an asterisk (*). Specify a single asterisk if, at OPEN time, you want to reject all volumes not defined to DFSMSrmm. You can use the installation exit, EDGUX100, to request that DFSMSrmm ignore such tapes.

Note: If you specify REJECT ANYUSE(*), all tapes not defined to the DFSMSrmm CDS cause an error message to be issued by DFSMSrmm when the tapes are used.

MEDINF commands

Use the MEDINF command to define media characteristics for OEM media products. The MEDINF values convert external values for media type and recording technology to internal values when supplied in the DFSMSrmm TSO subcommand input, as well as externalizes recorded values in reports and DFSMSrmm TSO subcommand output. To connect a volume to a specific media information entry, assign the media information name with the RMM TSO subcommands ADDVOLUME and CHANGEVOLUME using the MEDINF operand. The following values apply:

- **NAME(*medinf_name*)**

  Specifies a name for the media information. Specify a value between 1 and 8 alphanumeric characters. This value is used to match the recorded volume information to the MEDINF installation-defined media information. The values defined for MEDIATYPE, RECORDINGFORMAT, and CAPACITY are taken if the assigned media information for a volume matches *medinf_name*.

  There is no default.

- **MEDIATYPE(*mediatype_id*,*mediatype_name*)**

  Use this operand to identify the internally used *mediatype_id* to be converted to the externally used *mediatype_name* in reports and in the output of the DFSMSrmm TSO subcommands. Also, use this operand to identify the externally used *mediatype_name* to be converted to the internally used *mediatype_id* that is recorded in the control data set when used as input to RMM TSO subcommands.

  When you specify multiple MEDINF entries that use the same *mediatype_id*, but with different *mediatype_name* values, the first such MEDINF entry is used for the internal-to-external conversion. The remaining values are synonyms used for external-to-internal conversions.

  mediatype_id A number in the range from 1 to 255
  mediatype_name One to 8 alphanumeric or national characters

  There is no default. When you do not specify the MEDIATYPE operand, DFSMSrmm displays a *mediatype_id* of 0 and *mediatype_name* of *.

- **RECORDINGFORMAT(*recordingformat_id*,*recordingformat_name*)**

  Use this operand to identify the internally used *recordingformat_id* to be converted to the externally used *recordingformat_name* in reports and in the output of the DFSMSrmm TSO subcommands. Also, use this operand to identify the externally used *recordingformat_name* to be converted to the internally used *recordingformat_id* that is recorded in the control data set when used as input to RMM TSO subcommands.

  When you specify multiple MEDINF entries that use the same *recordingformat_id*, but with different *recordingformat_name* values, the first such MEDINF entry is used for the

internal-to-external conversion. The remaining values are synonyms used for external-to-internal conversions.

recordingformat_id A number in the range from 1 to 255
recordingformat_name One to 8 alphanumeric or national characters

There is no default. When you do not specify the RECORDINGFORMAT operand, DFSMSrmm displays a *recordingformat_id* of 0 and *recordingformat_name* of *.

- **CAPACITY(*capacity*)**

  Specifies the media capacity in MB available on the non-IBM media that has a media type *mediatype_id* and a recording format *recordingformat_id*. Specify a value from 0 to 4294967295.

  When you specify a capacity value, DFSMSrmm uses that value to set the capacity of a volume when the volume is added or when the recording format or media type of the volume is manually changed. When new data is written on a volume, DFSMSrmm uses your specified capacity only if the tape drive does not return the volume capacity.

  Do not specify the CAPACITY operand on a MEDINF command that specifies the same MEDINF NAME and MEDIATYPE *mediatype_id* and RECORDINGFORMAT *recordingformat_id* as an earlier MEDINF command. If you do, an information message EDG0243I is issued at start-up time, the CAPACITY operand is ignored, and the capacity from the earlier entry used instead. When you use synonyms, use the CAPACITY operand only when the combination of MEDINF NAME, MEDIATYPE *mediatype_id*, and RECORDINGFORMAT *recordingformat_id* are unique.

  Default: When you do not specify the CAPACITY operand, DFSMSrmm displays a capacity of 0.

- **REPLACE(PERM | TEMP | AGE | WMC)**

  Use this operand to identify the policies for replacement of volumes based on such values as read and write errors, age, and numbers of times written. Volumes are identified for replacement when one or more of the policy values are met or exceeded.

  Do not specify the REPLACE operand on a MEDINF command that specifies the same MEDINF NAME and MEDIATYPE *mediatype_id* and the RECORDINGFORMAT *recordingformat_id* as an earlier MEDINF command. If you do, an information message EDG0243I is issued at start-up time, the REPLACE operand is ignored, and the replace policy from the earlier entry is used instead. When you use synonyms, only use the REPLACE operand if the combination of MEDINF NAME, MEDIATYPE *mediatype_id* and RECORDINGFORMAT *recordingformat_id* are unique.

  When you add or modify a volume replacement policy, DFSMSrmm does not implement it retrospectively. Volumes that are already set with the REPLACE release action are not reprocessed and the action reset. In order to have DFSMSrmm reconsider the policy for all non-pending release volumes, you first have to manually change the REPLACE release action to SCRATCH, and then run EXPROC processing.

- **PERM(*count*)**

  Use this operand when your policy is to be based on permanent I/O errors. DFSMSrmm compares your value with the sum of permanent read and write errors over the life of the volume.

  The value range is 0 - 99999. Specify 0 to disable replacement based on permanent errors.

There is no default value.

- **TEMP(*count*)**

  Use this operand when your policy is to be based on temporary I/O errors. DFSMSrmm compares your value with the sum of temporary read and write errors over the life of the volume.

  The value range is 0 - 99999. Specify 0 to disable replacement based on temporary errors.

  There is no default value.

- **AGE(*years*)**

  Use this operand when your policy is to be based on the age of the volume. DFSMSrmm compares your value with the volume creation, and current date and time.

  The value range is 0 - 99999. Specify 0 to disable replacement based on age.

  There is no default value.

- **WMC(*count*)**

  Use this operand when your policy is to be based on how many times the volume is mounted for output and written to. DFSMSrmm compares your value with the write mount count for the volume.

  The valid value range is 0 - 65535. Specify 0 to disable replacement based on volume write mount count. Values in the range 65536 - 99999 are accepted, but have the same effect as specifying 0 (replacement based on volume write mount count is disabled).

  There is no default value.

OPENRULE commands

Use the OPENRULE command to identify special processing for DFSMSrmm to perform during OPEN processing. It applies equally to both system-managed and non-system-managed volumes.

In a shared CDS RMMplex, these commands allow you to control sharing volumes between systems while still maintaining a single CDS. OPENRULE allows you to specify what cannot be used as well as what can be used:

- **OPENRULE VOLUME(*) TYPE(NORMM) ANYUSE(IGNORE BY(ANY))**

  All volumes VOLUME(*) not defined in the DFSMSrmm control data set. TYPE(NORMM) can be used with the DFSMSrmm bypass processing keyword in your JCL (EXPDT=98000) if you have the necessary access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.* or STGADMIN.EDG.IGNORE.TAPE.* resource defined in the RACF class FACILITY.

- **OPENRULE VOLUME(*) TYPE(RMM) ANYUSE(ACCEPT)**

  This rule allows the normal use ANYUSE(ACCEPT) of all volumes VOLUME(*) defined to DFSMSrmm TYPE(RMM).

PRTITION commands

Use the PRTITION command to identify special processing for DFSMSrmm to perform during library entry, export/import, eject, and CUA processing, during OPEN processing, and EXPROC processing. The PRTITION commands apply to all volumes, not only

system-managed volumes. However, only system-managed volumes benefit from functions, such as library insert, CUA, and eject.

- `PRTITION VOLUME(*) TYPE(NORMM) SMT(IGNORE) NOSMT(IGNORE)`

  This command rejects all volume VOLUME(*) requests not defined in the DFSMSrmm CDS TYPE(RMM). There is no differentiation between system-managed tapes SMT(IGNORE) and non-system-managed tapes NOSMT(IGNORE).

SECCLS command

Use the SECCLS command to define security classes for data sets and volumes. These security classes appear in reports and RMM TSO subcommand output. DFSMSrmm records these security classes in the DFSMSrmm CDS only. It does not make RACF aware of them. There is no connection between these definitions and any similar definitions or function provided in RACF, but you can use similar values for overall consistency.

The examples of the SECCLS commands provided in Example 3-5 on page 96 show the use of the following functions:

- Volumes must be erased on release.
- Issue operator confirmation messages.
- Write SMF security records each time a tape data set in this class is opened.

MNTMSG command

Use the MNTMSG command to tailor mount and fetch messages so they display the VOLSER and pool name or rack number. This information makes it easier for the operator to pull and mount volumes.

> **Important:** Do not change the mount and fetch messages if you are using a vendor robotics system, such as an Oracle VSM.

VLPOOL command

Use the VLPOOL command to define pools. When you add a new volume to the library, DFSMSrmm assigns it a shelf location from the specified pool. The most important parameters are identified in the following list:

- **AUTOSCRATCH**

  Use this operand to override the scratch processing release action for volumes in this pool. By default, AUTOSCRATCH(YES) return to scratch is performed automatically when expiration processing is running on a system that has access to the catalogs, TCDB, and library for the volume. If you need to take any special actions not performed by DFSMSrmm, perhaps on another system, specify AUTOSCRATCH(NO).

  When you specify AUTOSCRATCH(YES) and do not manually confirm the scratch release action, DFSMSrmm performs return to scratch processing, including these options:

  - UNCATALOG parmlib option
  - TPRACF parmlib option
  - SMSTAPE(UPDATE(SCRATCH))

  When you specify AUTOSCRATCH(NO) or you manually confirm the scratch release action for any volume, DFSMSrmm does not perform this part of the return to scratch processing. DFSMSrmm assumes that you have manually performed the cleanup already.

  When you specify AUTOSCRATCH(NO), DFSMSrmm does not return the volume to scratch status until you have manually confirmed the volume release action and run expiration processing.

DFSMSrmm checks the AUTOSCRATCH setting and the scratch release action for the volume when the volume is returning to scratch. If the scratch release action is set and the volume is in a pool with AUTOSCRATCH(NO), DFSMSrmm leaves the volume in pending release status. You must perform whatever actions or cleanup activity you require and confirm that the scratch release action has been performed by using the subcommand: RMM CV *VOLSER* CRLSE(SCRATCH). Run expiration processing on any DFSMSrmm system to return the volume to scratch.

Default: AUTOSCRATCH(YES)

- **MASTEROVERWRITE**

  Specify the VLPOOL MASTEROVERWRITE operand to control how DFSMSrmm allows the overwriting of a volume. You can specify whether you want to allow data to be appended to the end of a volume or overwritten, or to allow new files to be added to a volume. The MASTEROVERWRITE value applies to all volumes that reside in a pool. If you do not specify a MASTEROVERWRITE value, DFSMSrmm uses the MASTEROVERWRITE value that you set using the EDGRMMxx parmlib member OPTION MASTEROVERWRITE operand, as described in “OPTION command” on page 79.

- **TYPE**

  In DFSMSrmm, you can specify one of two categories of pools:

  R A *rack pool* is shelf space that is assigned to hold volumes that are generally read-only, and that enter and leave your installation on an ad hoc basis. These volumes are typically software product volumes and customer volumes and do not adhere to your installation’s naming conventions.

  S A *scratch pool* is assigned to shelf space to hold all other volumes in the removable media library. The volumes that are assigned to this shelf space can be used to satisfy scratch requests as long as the volumes are in scratch status. After the volume has been written to, it becomes a master volume until the data is no longer required by the installation. The volume remains in the same DFSMSrmm scratch pool in that it occupies the same shelf space regardless of status.

  We advise that you define all the pools used in your tape management system when running in parallel.

- **RACF(N)**

  Specifies the type of RACF tape support that DFSMSrmm needs to provide for volumes in the pool you are defining.

  The following value applies:

  N Specify whether you do not want DFSMSrmm to create RACF tape profiles for the volumes in the pool.

  When you are defining RACF tape support for volumes in a pool, you must look at the RACF tape support you have defined for your installation with the OPTION TPRACF command.

Example 3-5 shows a sample EDGRMMxx PARMLIB member that you can use as a model when preparing for DFSMSrmm recording mode.

Example 3-5 Sample EDGRMMxx PARMLIB member

```
OPTION    OPMODE(R)                    /* Record-Only Mode         */-
          ACCOUNTING(J)                /* Accounting from JOB      */-
```

```
          BACKUPPROC(EDGBKUP)            /* Name of BACKUP-proc      */-
          BLP(RMM)                       /* DFSMSrmm controls BLP    */-
          CATRETPD(0012)                 /* catalog retention        */-
          CATSYSID(*)                    /* all catalogs shared      */-
          CDSID(PROD)                    /* control data set id      */-
          COMMANDAUTH(OWNER)             /* type of authorization    */-
          DATEFORM(J)                    /* Date format              */-
          DISPDDNAME(LOANDD)             /* DISP ctrl  DD card       */-
          DISPMSGID(EDG4054I)            /* DISP message number      */-
          DSNAME(RMM.PROD.CDS)           /* CDS data set name        */-
          EXPDTDROP(PERCENT(10,WARN))    /* Write warning msg only   */-
          GDG(CYCLEBY(GENERATION) DUPLICATE(BUMP))                   -
          IPLDATE(N)                     /* IPL date checking        */-
          JRNLNAME(RMM.PROD.JRNL)        /* JRNL data set name       */-
          JRNLTRAN(NO)                   /* Normal journaling        */-
          JOURNALFULL(75)                /* Percentage JRNL full     */-
          LINECOUNT(054)                 /* Lines per page           */-
          MASTEROVERWRITE(LAST)          /* Overwriting of a vol     */-
          MAXHOLD(100)                   /* Number of I/O oper.      */-
          MAXRETPD(NOLIMIT)              /* Maximum retention        */-
          MEDIANAME(3490)                /* Default medianame        */-
          MOVEBY(VOLUME)                 /* spec. how to move vols   */-
          MSG(M)                         /* case for message txt     */-
          NOTIFY(Y)                      /* Notify volume owners     */-
          PDA(OFF)                       /* PDA is disabled          */-
          PDABLKCT(255)                  /* number of blocks         */-
          PDABLKSZ(31)                   /* blocksize in K           */-
          PDALOG(OFF)                    /* PDA output disabled      */-
          PREACS(NO)                     /* Disable EDGUX100 ACS pr. */-
          RETAINBY(VOLUME)               /* spec. how to retain vols */-
          RETENTIONMETHOD(VRSEL)         /* for new tape data sets   */-
          RETPD(0005)                    /* Default retention        */-
          REUSEBIN(STARTMOVE)            /* reuse BIN as soon as pos.*/-
          SCRATCHPROC(EDGXPROC)          /* ATL/MTL procedure        */-
          SMFAUD(YES)                    /* SMF type 42-S22 audit rec*/-
          SMFSEC(YES)                    /* SMF type 42-S23 sec. rec */-
          SMSACS(YES)                    /* enable MV ACS processing */-
/*        SMSTAPE(UPDATE(EXITS,SCRATCH,COMMAND),PURGE(YES))       ATL*/-
          SYSID(EGZB)                    /* Name of the system       */-
          TPRACF(N)                      /* RACF tape support        */-
          TVEXTPURGE(EXPIRE(2))          /* set an expiration date   */-
          UNCATALOG(N)                   /* Catalog support          */-
          VRSCHANGE(INFO)                /* No additional action     */-
          VRSDROP(PERCENT(10,WARN))      /* Write warning msg only   */-
          VRSEL(NEW)                     /* New VRS processing       */-
          VRSJOBNAME(2)                  /* DATASETNAME/JOBNAME      */-
          VRSMIN(0000000100,WARN)        /* Warn if < 100 VRSs       */-
          VRSRETAIN(PERCENT(80,WARN))    /* Write warning msg only   */
/* ******************************************************************* */
MNTMSG    MSGID(IEF233A)      ID(001) VOLUME(016) RACK(016)
MNTMSG    MSGID(IEF233D)      ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEF234E K')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEF234E R')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEF234E D')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID(IEF455D)      ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID(IEC501A)      ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEC502E K')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEC502E D')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEC502E R')  ID(001) VOLUME(016) RACK(999)
MNTMSG    MSGID('IEC502E RD') ID(001) VOLUME(017) RACK(999)
```

```
MNTMSG    MSGID('IEC502E RK') ID(001) VOLUME(017) RACK(999)
MNTMSG    MSGID(IAT5110)      ID(001) VOLUME(044) RACK(999)
MNTMSG    MSGID(IAT5210)      ID(001) VOLUME(050) RACK(999)
MNTMSG    MSGID(IAT5410)      ID(001) VOLUME(020) RACK(999)
/* ************************************************************** */
/*  MEDINF command added at 2008/03/21         Norbert           */
/* ************************************************************** */
MEDINF NAME(VTFM)                        /* VTFM global replacement */-
       REPLACE(PERM(0) WMC(0) TEMP(0) AGE(0))
MEDINF NAME(VTFM)                        /* define the VTFM volumes */-
       MEDIATYPE(5,ETC)                  /* standard definition     */-
       CAPACITY(1000)
MEDINF NAME(VTFM)                        /* define the VTFM volumes */-
       MEDIATYPE(5,ETC)                                             -
       RECORDINGFORMAT(5,EFMT1)                                     -
       CAPACITY(1000)
/* ************************************************************** */
/*  REJECT command removed at 2008/03/21        Norbert           */
/* ************************************************************** */
/* REJECT    ANYUSE(*)        Replaced by the following OPENRULE and */
/*                            PRTITION commands                    */
/* ************************************************************** */

/* ************************************************************** */
/*  PRTITION command added at 2008/03/21        Norbert           */
/* ************************************************************** */
PRTITION VOLUME(*)              /*     added 2008/03/21 NS         */-
         TYPE(NORMM)            /* volumes not defined in CDS      */-
         SMT(IGNORE)            /* system-managed                  */-
         NOSMT(IGNORE)          /* non-system-managed              */
/* ************************************************************** */
/*  OPENRULE commands are added at 2008/03/21          Norbert     */
/* ************************************************************** */
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS         */-
         TYPE(NORMM)            /* volumes not defined in CDS      */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing           */
                                /*   ignore all types of request   */
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS         */-
         TYPE(RMM)              /* all volumes defined in CDS      */-
            ANYUSE(ACCEPT)      /* allow OPEN processing           */
/* ************************************************************** */
VLPOOL    PREFIX(NS*)                /* volume prefix             */ -
          AUTOSCRATCH(NO)            /* do not return to scratch  */ -
          TYPE(R)                    /* do not satisfy scratch req */ -
          RACF(N)                    /* SAF tape security is inuse */ -
          MASTEROVERWRITE(USER)      /* allow to overwrite DSNs   */ -
          MEDIANAME(3590)            /* media name used           */ -
          EXPDTCHECK(Y)              /* check VOL1 expdt          */ -
          DESCRIPTION('Norberts scratch pool')
VLPOOL    PREFIX(THS*)               /* volume prefix             */ -
          AUTOSCRATCH(NO)            /* do not return to scratch  */ -
          TYPE(S)                    /* satisfy scratch requests  */ -
          RACF(N)                    /* SAF tape security is inuse */ -
          MEDIANAME(3592)            /* media name used           */ -
          EXPDTCHECK(N)              /* do not check VOL1 expdt   */ -
          DESCRIPTION('3592 POOL')
VLPOOL    PREFIX(VT*)                /* volume prefix             */ -
          AUTOSCRATCH(YES)           /* return to scratch auto    */ -
          DESCRIPTION('VTFM 3592 POOL')                            -
          EXPDTCHECK(N)              /* do not check VOL1 expdt   */ -
```

```
          MASTEROVERWRITE(ADD)        /* ctrl DSN overwriting       */ -
          MEDIANAME(3592)             /* media name used            */ -
          NAME($$DFLT)                /* pool name                  */ -
          RACF(N)                     /* Type of RACF tape support  */ -
          TYPE(S)                     /* satisfy scratch requests   */
/* ************************************************************** */
LOCDEF    LOCATION(SANJOSE)                                          -
          TYPE(STORAGE)                                              -
          PRIORITY(2000)                                             -
          MEDIANAME(*)                                               -
          MANAGEMENTTYPE(BINS)
LOCDEF    LOCATION(IBMATL1)                                          -
          TYPE(LIBRARY)                                              -
          PRIORITY(4800)
LOCDEF    LOCATION(VTFM001)                                          -
          AUTOMOVE(NO)                                               -
          MANAGEMENTTYPE(NOBINS)                                     -
          MEDIANAME(*)                                               -
          TYPE(STORAGE,HOME)
/* ************************************************************** */
SECCLS    NUMBER(010)                                                -
          NAME(UCL)                                                  -
          SMF(N)                                                     -
          MESSAGE(N)                                                 -
          ERASE(N)                                                   -
          DESCIPTION('UNCLASSIFIED')                                 -
          MASK('**')
SECCLS    NUMBER(050)                                                -
          NAME(IUO)                                                  -
          SMF(Y)                                                     -
          MESSAGE(N)                                                 -
          ERASE(N)                                                   -
          DESCIPTION('INTERNAL USE ONLY')                            -
          MASK('**.IUO.**',                                          -
               '**.INTERNAL.**',                                     -
               '**.INT.**')
SECCLS    NUMBER(100)                                                -
          NAME(IC)                                                   -
          SMF(Y)                                                     -
          MESSAGE(Y)                                                 -
          ERASE(Y)                                                   -
          DESCIPTION('EXTERNAL USE ONLY')                            -
          MASK(**.IC.**)
```

COMMNDxx (optional): Commands automatically issued at initialization

Update COMMNDxx to automatic commands that the system internally issues as part of system initialization. COMMNDxx is useful for automatic entry of commands that are frequently issued at system initialization.

To start the DFRMM-started task at system initialization, add the highlighted statement in Figure 3-18 on page 100.

```
...
COM='S VTAM,,,(LIST=&NODE;)'
COM='S NETVIEW,SUB=MSTR,CAT=&CATALOG;,SYS=&SYSNAME;'
COM='S APPC,SUB=MSTR,APPC=02'
COM='S ICR&SYSNAME;.SYSLOG'
COM='SET DAE=01'
COM='SET DAE=&DAE;'
COM='SETXCF START,POLICY,TYPE=CFRM,POLNAME=&POLNAME;'
COM='S DFRMM,M=70'
...
```

Figure 3-18 COMMNDxx to start a DFRMM-started task at initialization

SMFPRMxx (optional): SMF parameters

If you want security or audit records, update SMFPRMxx to define the SMF record numbers that DFSMSrmm generates. In DFSMSrmm, you define the SMF record numbers in the EDGRMMxx PARMLIB member using two operands: OPTION SMFAUD for auditing records and OPTION SMFSEC for security records. For more information about these commands, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Select two SMF record numbers in the 128 - 255 range that your installation is not using. Add your SMFAUD and SMFSEC record numbers to SMFPRMxx as specified in *z/OS MVS System Management Facilities (SMF)*, SA22-7630. By default, there is no SMF recording.

You can use the MVS command DISPLAY SMF as shown in Figure 3-19 to display SMF data.

```
DISPLAY SMF,O
```

Figure 3-19 Display SMF information

Note: The 0 directs the system to display the current SMF options.

Example 3-6 shows the output of the display SMF command.

Example 3-6 Output of a DISPLAY SMF command

```
IEE967I 18.39.10 SMF PARAMETERS 982
        MEMBER = SMFPRM00
        SMFDLEXIT(USER3(IRRADU86)) -- DEFAULT
        SMFDLEXIT(USER2(IRRADU00)) -- DEFAULT
        SMFDPEXIT(USER3(IRRADU86)) -- DEFAULT
        SMFDPEXIT(USER2(IRRADU00)) -- DEFAULT
        EMPTYEXCPSEC(NOSUPPRESS) -- DEFAULT
        MULCFUNC -- DEFAULT
        BUFUSEWARN(25) -- DEFAULT
        BUFSIZMAX(0128M) -- DEFAULT
        MAXEVENTINTRECS(00) -- DEFAULT
        SYNCVAL(00) -- DEFAULT
        DUMPABND(RETRY) -- DEFAULT
        SUBSYS(STC,NOINTERVAL) -- SYS
        SUBSYS(STC,NODETAIL) -- SYS
        SUBSYS(STC,EXITS(IEFUS0)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFUJP)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFUJI)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFACTRT)) -- PARMLIB
```

```
SUBSYS(STC,EXITS(IEFU85)) -- PARMLIB
SUBSYS(STC,EXITS(IEFU84)) -- PARMLIB
SUBSYS(STC,EXITS(IEFU83)) -- PARMLIB
SUBSYS(STC,EXITS(IEFU29)) -- PARMLIB
SUBSYS(STC,TYPE(0:18,20:98,100:255)) -- PARMLIB
SYS(NODETAIL) -- PARMLIB
SYS(NOINTERVAL) -- PARMLIB
SYS(EXITS(IEFU29)) -- PARMLIB
SYS(EXITS(IEFUTL)) -- PARMLIB
SYS(EXITS(IEFUJI)) -- PARMLIB
SYS(EXITS(IEFUSO)) -- PARMLIB
SYS(EXITS(IEFUJP)) -- PARMLIB
SYS(EXITS(IEFUSI)) -- PARMLIB
SYS(EXITS(IEFUJV)) -- PARMLIB
SYS(EXITS(IEFACTRT)) -- PARMLIB
SYS(EXITS(IEFU85)) -- PARMLIB
SYS(EXITS(IEFU84)) -- PARMLIB
SYS(EXITS(IEFU83)) -- PARMLIB
SYS(TYPE(0:18,20:98,100:255)) -- PARMLIB
NOBUFFS(MSG) -- PARMLIB
LASTDS(MSG) -- PARMLIB
LISTDSN -- PARMLIB
SID(SC70) -- PARMLIB
DDCONS(NO) -- PARMLIB
JWT(2400) -- PARMLIB
MEMLIMIT(NOLIMIT) -- PARMLIB
STATUS(010000) -- PARMLIB
MAXDORM(3000) -- PARMLIB
INTVAL(10) -- PARMLIB
REC(PERM) -- PARMLIB
NOPROMPT -- PARMLIB
DSNAME(SYS1.SC70.MAN3) -- PARMLIB
DSNAME(SYS1.SC70.MAN2) -- PARMLIB
DSNAME(SYS1.SC70.MAN1) -- PARMLIB
ACTIVE -- PARMLIB
```

You can use the MVS SET SMF operator command (see Figure 3-20) to modify the SMF recording options dynamically by specifying which SMFPRMxx parmlib member is to be used.

Using the SET command, the installation can replace all the existing SMF options. For example, an installation can activate SMF recording after an IPL in which NOACTIVE is specified by using the SET command and choosing the parmlib member that contains the ACTIVE option. The SET command, however, cannot change the SID parameter.

```
SET SMF=xx
```

Figure 3-20 SET SMF command

The two alphanumeric characters indicate the SMFPRMxx member of the logical parmlib that contains the parameters the system is to use when restarting SMF.

GRSRNLxx (optional): GRS resourcename list

If you are not using global resource serialization (GRS) to control enqueues among sharing systems, you do not have to update GRSRNLxx. If you have multiple systems that are connected in a GRSplex, decide how you want GRS to handle the reserve on the CDS.

Restriction: The length of RNAME must be exactly 23 characters. If the length of your CDSID is fewer than 8 characters, we advise that you use a generic entry as described here. Otherwise, you must specify RNAME as a hexadecimal constant:

```
RNAME(X'D4C1E2E3C5D94BD9C5E2C5D9E5C54Bxx..xx4040')
```

`X'D4C1E2E3C5D94BD9C5E2C5D9E5C54B'` represents MASTER.RESERVE, and `X'xx..xx4040'` represents your CDSID appended with blanks to 8 characters in length.

If the volume where the CDS resides does not contain other critical data, and real reserves will not cause any shared disk contention problems, add the reserve name to the GRS exclusion list. This leaves the real hardware reserve in effect and causes the associated SYSTEMS enqueue to be converted to a local SYSTEM enqueue. The delay that is associated with sending the additional unnecessary enqueue around the GRSplex is then avoided.

The required statements in GRSRNLxx to convert the SYSTEMS enqueue to a local SYSTEM enqueue are shown in Figure 3-21.

```
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
       QNAME(SYSZRMM)
       RNAME(MASTER.RESERVE)
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
       QNAME(SYSZRMM)
       RNAME(X'D4C1E2E3C5D94BD9C5E2C5D9E5C54BD7D9D6C4404040 40')
```

Figure 3-21 GRS EXCL definition TYPE(SPECIFIC) with a hexadecimal constant

Important: You must define both resources if you are using DFSMSrmm release z/OS V1.9 or higher and you have specified TYPE(SPECIFIC). The first time DFSMSrmm starts, it needs to reserve on MASTER.RESERVE and initialize RMM CDSID support.

If the volume where the CDS resides contains other critical data, and real reserves can affect other systems or cause disk lockouts, add the reserve name to the GRS reserve conversion list. This leaves the global SYSTEMS ENQUEUE in effect and removes the real hardware reserve.

Figure 3-22 shows the required statements in GRSRNLxx to convert the RESERVE to a SYSTEMS enqueue.

```
RNLDEF RNL(CON) TYPE(GENERIC)
       QNAME(SYSZRMM)
       RNAME(MASTER.RESERVE)
```

Figure 3-22 GRS CON definition with TYPE(GENERIC)

Note: If you are not using GRS, and use an equivalent product instead, use the enquiry character (ENQ) resource name information that is provided in the DFSMSrmm publications to define the resource correctly to the other product. Do not use generic or global options for SYSZRMM. You must ensure that each resource with the SYSZRMM major name is processed correctly.

Figure 3-23 shows the difference between a system enqueue and a hardware reserve.

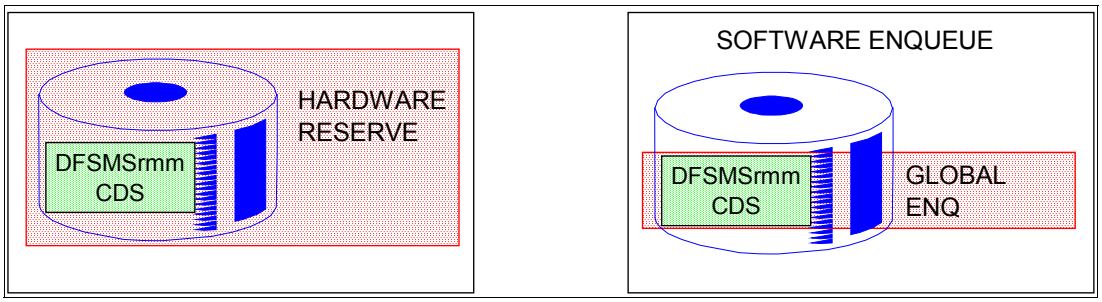


Figure 3-23 Use of GRS enqueue and hardware reserve

You can use the MVS SET GRSRNL command shown in Figure 3-24 to change the current GRSRNL definitions.

```
SET GRSRNL=(xx[,xx]...)
```

Figure 3-24 MVS SET GRSRNL command

Important: Use care when you enter the SET GRSRNL command to change heavily used or highly critical resources. Work that requires resources for a critical application, or resources used by the operating system might become suspended or delayed, which can impair the performance of that critical application or the operating system itself.

The two alphanumeric characters indicate the GRSRNLxx members of the logical parmlib that contains the specified GRS resourcename lists. GRSRNL allows you to change the current Resource Name Lists (RNLs) specified in one or more GRSRNLxx parmlib members. Do not use parentheses when only one parmlib member is specified.

You cannot specify the GRSRNL= parameter if GRSRNL has already been set to EXCLUDE in the logical parmlib member IEASYSxx.

The GRSRNL= parameter has a sysplex scope only when all systems share the same parmlib data set.

Figure 3-25 shows how you can use the MVS DISPLAY GRS command to display your current GRS information.

```
DISPLAY GRS,ALL
```

Figure 3-25 MVS DISPLAY GRS command

Example 3-7 contains the output of the DISPLAY GRS command.

Example 3-7 DISPLAY GRS messages

```
RESPONSE=SC70
 ISG343I 18.58.51 GRS STATUS 997
      LIST TYPE QNAME    RNAME
      INCL GEN  SPFEDIT
      INCL GEN  SYSDSN
      INCL GEN  SYSIKJBC
      INCL GEN  SYSIKJUA SYS1
```

```
      INCL SPEC SYSIKJUA VAIN
      INCL GEN  SYSZVOLS
      EXCL SPEC SPFEDIT  PASSWORD
      EXCL SPEC SPFEDIT  SYS1.DCMLIB
      EXCL SPEC SPFEDIT  SYS1.NUCLEUS
      EXCL SPEC SPFEDIT  SYS1.SVCLIB
      EXCL GEN  SYSCTLG
      EXCL SPEC SYSDSN   PASSWORD
      EXCL SPEC SYSDSN   SYS1.DCMLIB
      EXCL GEN  SYSDSN   SYS1.JES3
      EXCL SPEC SYSDSN   SYS1.NUCLEUS
      EXCL SPEC SYSDSN   SYS1.SVCLIB
      EXCL SPEC SYSIGGV2 CATALOG.SHRICF1.VIODFBK
      EXCL SPEC SYSIGGV2 CATALOG.SHRICF1.VIODFPK
      EXCL SPEC SYSIGGV2 CATALOG.TOTICF1.VO33SMP
      EXCL SPEC SYSIGGV2 CATALOG.TOTICF1.VO33SM1
      EXCL SPEC SYSIGGV2 UCAT.VBOOK01
      EXCL SPEC SYSIGGV2 UCAT.VOS3R2B
      EXCL GEN  SYSZJES2
      CON  GEN  ARCGPA
      CON  GEN  DSPURI01
      CON  SPEC HWRESERV BOOK01
      CON  SPEC HWRESERV IODFBK
      CON  SPEC HWRESERV IODFPK
      CON  SPEC HWRESERV OS3R2B
      CON  SPEC HWRESERV O33SMP
      CON  SPEC HWRESERV O33SM4
      CON  GEN  IGDCDSXS
      CON  GEN  SPFEDIT
      CON  GEN  SPZAPLIB
      CON  GEN  SYSIEWLP
      CON  GEN  SYSIGGV2
      CON  GEN  SYSVTOC
      CON  GEN  SYSZRACF
      CON  SPEC SYSZRMM  MASTER.RESERVE
      CON  GEN  SYSZVVDS
NO ENQ RESOURCE CONTENTION EXISTS
NO REQUESTS PENDING FOR ISGLOCK STRUCTURE
NO LATCH CONTENTION EXISTS
SYSTEM    STATE              SYSTEM    STATE
SC63      CONNECTED          SC65      CONNECTED
SC70      CONNECTED          SC64      CONNECTED
THERE IS NO RNL CHANGE IN PROGRESS.
GRS STAR MODE INFORMATION
  LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
  THE CONTENTION NOTIFYING SYSTEM IS SC63
  SYNCHRES:      YES
  ENQMAXU:     16384
  ENQMAXA:    250000
  GRSQ:   CONTENTION
```

Table 3-4 shows the DFSMSrmm resource symbolic names. Do not use GRS to change or alter any of these ENQs. The resource symbolic names are provided for information only because DFSMSrmm processing manages serialization for you.

Table 3-4 DFSMSrmm resource symbolic names

Major name	Minor name	Resource	Scope
SYSZRMM	HSKP.*dsn.VOLSER*	Inventory management data set serialization	SYSTEMS

Major name	Minor name	Resource	Scope
SYSZRMM	MASTER.RESERVE	DFSMSrmm control data set serialization at start-up and when the CDSID is not yet known	SYSTEMS
SYSZRMM	MASTER.RESERVE.*cdsid*	DFSMSrmm control data set serialization	SYSTEMS
SYSZRMM	MHKP.ACTIVE	Serialize inventory management functions on the same DFSMSrmm subsystem	SYSTEMS
SYSZRMM	MHKP.*dsn*.*VOLSER*	Inventory management data set serialization	SYSTEMS
SYSZRMM	RMM.ACTIVE	Ensure that only one system is run per z/OS image	SYSTEMS
SYSZRMM	BUFFER.CONTROL	Buffer management	STEP
SYSZRMM	EDGINERS.*VOLSER*	Serialize volume labeling	SYSTEMS
SYSZRMM	SHUTDOWN	Serialize DFSMSrmm shutdown and refresh processing	SYSTEMS
SYSZRMM	INACTIVE	Serialize DFSMSrmm activation that enables only a single write to operator with reply (WTOR) to be issued to the operator	SYSTEMS
SYSZRMM	EXIT_IS_ACTIVE	Exit recovery serialization	SYSTEMS
SYSZRMM	WTOR_ENQ	Exit recovery serialization	SYSTEMS
SYSZRMM	WTORIPC	TCP/IP error recovery serialization on CLIENT systems	SYSTEMS
SYSZRMM	EXIT_*id*_UNAVAIL	Exit recovery serialization where id can be 100, 200, or 300 representing the last three characters of the DFSMSrmm installation exits EDG_EXIT100, EDG_EXIT200, or EDG_EXIT300	SYSTEMS

Setting up MIM multi-image integrity for DFSMSrmm

If you installed the Computer Associates International, Inc., product Multi-Image Manager (CA-MIM) or MIM as opposed to GRS, you must update your MIM definitions. DFSMSrmm uses the RESERVE macro and ENQ macro to serialize use of its own resources. The SCOPE that is specified by DFSMSrmm ensures that serialization is across systems, within a system, or within a task, as required. All serialized resources use the SYSZRMM QNAME, and the RNAME is determined based on the resource required. If you do nothing to define the SYSZRMM QNAME, the serialization that is required by DFSMSrmm needs to be provided.

For a complete list of the resources that are serialized for DFSMSrmm, and the RNAMES, and SCOPE of each request, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

To serialize the DFSMSrmm CDS, the RESERVE macro with an RNAME of MASTER.RESERVE is used, which results in the following objects:

- A hardware reserve
- A SYSTEMS enqueue

In a multisystem environment, the SYSTEMS enqueue is not required nor is a hardware reserve. There might also be times when a hardware reserve is not required because of other

data on the same volume. You can optionally use MIMQNAME member entries to optimize the DFSMSrmm serialization for your environment.

If you want to use hardware reserves, we suggest the MIMQNAME member of the CNTL data set shown in Figure 3-26.

```
SYSZRMM    GDIF=YES,        /*  GDIF SHOULD PROCESS THIS QNAME
           SCOPE=SYSTEMS,   /*  Identify which ENQUEUES to manage
           RESERVES=KEEP,   /*  Use Hardware reserves if requested
           EXEMPT=YES,      /*  EXEMPT LIST is required for SCOPE=SYSTEMS
           ECMF=NO,         /*  ECMF not required
           TRACE=NONE       /*  Do not trace requests
```

Figure 3-26 MIMQNAME member using hardware reserves

The SCOPE=SYSTEMS ENQs that are issued by DFSMSrmm need to be propagated to ensure correct serialization, except for the ENQ related to the hardware reserve. The EXEMPT=YES identifies that one ENQ must not be propagated. You specify these in the GDIEXMPT member (see Figure 3-27).

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*  DFSMSrmm LOCAL EXCLUSION STATEMENTS. DO NOT PROPAGATE ENQS       *
*  FOR CDS SERIALIZATION.                                           *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
LOCAL    QNAME=SYSZRMM,RNAME=MASTER.RESERVE
LOCAL    QNAME=SYSZRMM,RNAME=MASTER.RESERVE.PROD
```

Figure 3-27 GDIEXMPT member

Important: You must define both resources if you are using DFSMSrmm release z/OS V1.9 or higher. The first time DFSMSrmm starts, it needs to reserve on MASTER.RESERVE and initialize RMM CDSID support.

If you do not want to use hardware reserves, and rely on propagation of the ENQs that serialize the DFSMSrmm CDS, we suggest the MIMQNAME member of the CNTL data set that is shown in Figure 3-28.

```
SYSZRMM   GDIF=YES,          /* GDIF SHOULD PROCESS THIS QNAME
          SCOPE=SYSTEMS,     /* Identify which ENQUEUES to manage
          RESERVES=CONVERT,  /* Use Hardware reserves if requested
          EXEMPT=YES,        /* EXEMPT LIST is required for SCOPE=SYSTEMS
          ECMF=NO,           /* ECMF not required
          TRACE=NONE         /* Do not trace requests
```

Figure 3-28 MIMQNAME member using propagation of ENQs

You can optionally specify other keyword values, such as ECMF=YES or TRACE=CONFLICT or TRACE=ALL, depending on your local requirements and whether you are testing with DFSMSrmm or running in parallel.

DEVSUPxx (optional): Device support options

DFSMS V1.8 and higher provide options for securing tape data sets using System Authorization Facility (SAF). These are designed to allow you to define profiles to protect data sets on tape using the DATASET class without the need to activate the TAPEDSN option or the TAPEVOL class. DFSMS also provides options that you can use to specify that all data sets on a tape volume need to have common authorization and that users are authorized to overwrite existing files on a tape volume.

For optimum tape security, use the combined capabilities of DFSMSrmm, DFSMSdfp, and RACF. It is suggested that you specify the parameters in your DEVSUPxx parmlib member as shown in Figure 3-29.

```
TAPEAUTHDSN=YES,
TAPEAUTHF1=YES,
TAPEAUTHRC4=FAIL,
TAPEAUTHRC8=FAIL
```

Figure 3-29 DEVSUPnn sample

The following values apply:

TAPEAUTHDSN Enables tape authorization checks in the DATASET class but without DSTYPE=T.

DSTYPE=T indicates to RACF that the check is for a data set on a tape volume and that special RACF tape data set and tape volume processing are to be performed. Without DSTYPE=T, RACF authorization checking considers only profiles in the DATASET class. The system uses the data set name that is specified in the allocation or JCL to check your authorization to read or write the specified file. In addition, the system determines the RACF erase-on-scratch setting from the RACF profile and passes it to your tape management system. Use this option only when you have a tape management system, such as DFSMSrmm, installed and actively checking that the 44-character data set name specified by the user matches the data set name on tape. Without a tape management system, tape data set open processing can only validate the last 17 characters of the data set name against the tape volume labels.

When you request BLP and the mounted volume uses standard labels, OPEN issues the authorization check that the user is authorized to use BLP. This processing uses the existing ICHBLP resource in the RACF FACILITY class. When you specify TAPEAUTHDSN=YES only, it replaces the check that RACF makes as part of tape volume authorization checking.

TAPEAUTHF1 Enables additional tape authorization checks in the DATASET class for existing files on the same tape volume when any other file on the tape volume is opened.

This function depends on the tape management system returning the 44-character data set name and data set sequence number to OPEN/EOV through the IFGTEP during the Volume Mount exit Volume Security function. If no data set name is returned by the tape management system, processing is as if this keyword had not been specified.

Although intended to enable an additional authorization check for the first data set when any other data set on the tape volume is opened, the implementation allows your tape management system to request one or more additional authorization checks when any data set on a tape volume is opened. Each additional data set name and data set sequence number returned results in an additional RACROUTE. Do not use this function unless you have a tape management system and it can return a data set name and data set sequence number. A data set sequence number is the label number that is normally specified in the JCL LABEL keyword and stored in the catalog.

When TAPEAUTHDSN=YES is in use, any additional RACROUTE matches that issued for TAPEAUTHDSN, except for the data set name and data set sequence number. Otherwise, TAPEAUTHF1 uses a RACROUTE that matches that used for SETROPTS TAPEDSN. When neither TAPEAUTHDSN nor SETROPTS TAPEDSN is in use, TAPEAUTHF1 support is not provided.

TAPEAUTHRC4 Denies accessing of data sets that are not protected by a security profile.

TAPEAUTHRC8 Denies accessing of data sets that typically cannot be accessed.

The combination of DFSMSrmm, DFSMSdfp, and RACF ensures the following functions:

- Full 44-character data set name validation.
- Validation that the correct volume is mounted.
- Control the overwriting of existing tape data sets.
- Management of tape data set retention.
- Control over the creation and destruction of tape volume labels.
- No limitations are caused by RACF TAPEVOL profile sizes and TVTOC limitations.
- All tape data sets on a volume have a common authorization.
- Use of generic DATASET profiles, enabling common authorization with DASD data sets.
- Authorization for all tape data sets regardless of the tape label type.
- Authorization for the use of BLP.
- Exploitation of RACF “erase on scratch” support.
- Use of DFSMSrmm FACILITY class profiles for data sets unprotected by RACF.
- Your authorization to use a volume outside of DFSMSrmm control through “ignore” processing also enables authorization to the data sets on that volume.

Use the MVS SET DEVSUP command that is shown in Figure 3-30 to implement the new tape data set security settings.

```
/SET DEVSUB=70
```

or

```
/T DEVSUB=70
```

Figure 3-30 Update the tape security settings

Example 3-8 shows the successful result of the command.

Example 3-8 SET DEVSUB messages

```
T DEVSUP=00
IEE252I MEMBER DEVSUP00 FOUND IN SYS1.PARMLIB
IEE536I DEVSUP   VALUE 00 NOW IN EFFECT
IEA253I DEVSUP  3480X RECORDING MODE DEFAULT IS COMPACTION.
```

```
IEA253I DEVSUP  ISO/ANSI TAPE LABEL VERSION DEFAULT IS V3
IEA253I DEVSUP  TAPE OUTPUT DEFAULT BLOCK SIZE LIMIT IS 32760
IEA253I DEVSUP  COPYSDB DEFAULT IS INPUT
IEA253I DEVSUP  STORAGE LIMIT FOR TAPE DDR SWAP DEFAULTED TO 1000M
IEA253I DEVSUP  TAPEAUTHDSN: YES
IEA253I DEVSUP  TAPEAUTHF1: YES
IEA253I DEVSUP  TAPEAUTHRC4: ALLOW
IEA253I DEVSUP  TAPEAUTHRC8: WARN
IEA253I DEVSUP  PERFORM NORMAL EXPIRATION DATE PROCESSING
```

Note: Only the tape security setting is updated. All other values, such as the media categories, are not updated.

For more information, see Chapter 13, “System Authorization Facility tape security” on page 475.

IGDSMSxx (optional): Storage management subsystem definition

Update the storage management subsystem definition member to set the CA_RECLAIM DATACLAS value. This parameter specifies whether to use CA reclaim for key-sequenced data sets (KSDSs) according to the CA Reclaim attribute in their data classes.

DATACLAS indicates that the SMS-managed KSDSs and non-SMS-managed KSDSs will go by the data class specification of CA Reclaim=Y|N at definition time or a subsequent ALTER setting. Only the resultant catalog setting of CA Reclaim=Y will perform CA reclaim; CA Reclaim=N will not perform CA reclaim. Use this setting to inform a KSDS not to use CA reclaim.

Note: If CA_RECLAIM is in effect, there is no longer a need to reorganize your Virtual Storage Access Method (VSAM) KSDS DFSMSrmm control data set.

```
CA_RECLAIM(DATACLAS)
```

Figure 3-31 CA_RECLAIM command syntax

Use the MVS DISPLAY SMS command to check the current SMS settings as shown in Figure 3-32.

```
DISPLAY SMS,OPTIONS
```

Figure 3-32 MVS DISPLAY SMS command

Example 3-9 contains the output of the DISPLAY SMS command.

Example 3-9 DISPLAY SMS messages

```
RESPONSE=SC70
 IGD002I 17:55:33 DISPLAY SMS 201
 ACDS     = SYS1.SMS.ACDS.NEW
 COMMDS   = SYS1.SMS.COMMDS
 ACDS LEVEL = z/OS V1.13
 INTERVAL = 15           DINTERVAL = 150
 SMF_TIME = YES          CACHETIME = 3600
 CF_TIME = 1800          PDSE_RESTARTABLE_AS = YES
```

```
PDSE_BMFTIME = 3600     PDSE1_BMFTIME = 3600
PDSE_LRUTIME = 60       PDSE1_LRUTIME = 50
PDSE_LRUCYCLES = 15    PDSE1_LRUCYCLES = 200
LOCAL_DEADLOCK = 15     GLOBAL_DEADLOCK = 4
REVERIFY = NO           DSNTYPE = PDS
ACSDEFAULTS = NO        PDSESHARING = EXTENDED
OVRD_EXPDT = NO         SYSTEMS = 8
PDSE_HSP_SIZE = 0MB   PDSE1_HSP_SIZE = 256MB
USE_RESOWNER = YES      RLS_MAX_POOL_SIZE = 100MB
RLSINIT = YES           RLSTMOUT = 0
COMPRESS = GENERIC      LOG_OF_LOGS = IGWTVS.LOG.OF.LOGS
QTIMEOUT = 300          TVSNAME = 070
AKP = 1000              TV_START_TYPE = WARM
MAXLOCKS = (0,0)
CICSVR_INIT = NO        CICSVR_DSNAME_PREFIX = DWWUSER.V3R1M0
CICSVR_RCDS_PREFIX = DWW
CICSVR_GRPNAME_SUFFIX = PROD
CICSVR_ZZVALUE_PARM =
CICSVR_UNDOLOG_CONTROL =
CICSVR_UNDOLOG_PREFIX = DWW
CICSVR_RCDS_PREFIX = DWW
CICSVR_GRPNAME_SUFFIX = PROD
CICSVR_ZZVALUE_PARM =
CICSVR_UNDOLOG_CONTROL =
CICSVR_UNDOLOG_PREFIX = DWW
CICSVR_BACKOUT_CONTROL =
CICSVR_GENERAL_CONTROL =
Rls_MaxCfFeatureLevel = Z
RlsAboveThebarMaxPoolSize = 0  RlsFixedPoolSize = 0
PDSE_MONITOR = (YES,0,0) PDSE1_MONITOR = (YES,0,0)
PDSE_DIRECTORY_STORAGE = 2000M PDSE1_DIRECTORY_STORAGE = 2000M
PDSE_BUFFER_BEYOND_CLOSE = NO   PDSE1_BUFFER_BEYOND_CLOSE = NO
GDS_RECLAIM = YES       DSSTIMEOUT = 0
BLOCKTOKENSIZE = NOREQUIRE       FAST_VOLSEL = ON
USEEAV = YES            BREAKPOINTVALUE = 21
OAMPROC =               SUPPRESS_DRMSGS = NO
OAMTASK =           PDSE_SYSEVENT_DONTSWAP = NO
DB2SSID = D9DG           SAM_USE_HPF = YES
CA_RECLAIM = DATACLAS
IGD002I 17:55:33 DISPLAY SMS
TRACE    = ON     SIZE = 128K     TYPE = ALL
  JOBNAME = *    ASID = *
  TRACING EVENTS:
    MODULE = ON   SMSSJF = ON   SMSSSI = ON   ACSINT = ON
    OPCMD  = ON   CONFC  = ON   CDSC   = ON   CONFS  = ON
    MSG    = ON   ERR    = ON   CONFR  = ON   CONFA  = ON
    ACSPRO = ON   IDAX   = ON   DISP   = ON   CATG   = ON
    VOLREF = ON   SCHEDP = ON   SCHEDS = ON   VTOCL  = ON
    VTOCD  = ON   VTOCR  = ON   VTOCC  = ON   VTOCA  = ON
    RCD    = ON   DCF    = ON   DPN    = ON   TVR    = ON
    DSTACK = ON   UAFF   = ON   DEBUG  = ON
VOLSELMSG = (OFF,0)    TYPE = ALL    JOBNAME = *
    ASID = *  STEPNAME = *
    DSNAME = *
```

When CA_RECLAIM is not in use, you can use the MVS SETSMS command to change the setting as shown in Figure 3-33 on page 111.

```
SETSMS CA_RECLAIM(DATACLAS)
```

Figure 3-33 MVS SETMVS command syntax

If the CA_RECLAIM value is changed, you need to get an additional message as shown in Figure 3-34.

```
IGW467I DFSMS CA_RECLAIM PARMLIB VALUE CHANGED ON SYSTEM:  SC64 854
OLD VALUE:   NONE
NEW VALUE:   DATACLAS
IEE712I SETSMS   PROCESSING COMPLETE
```

Figure 3-34 SETSMS messages

If you have specified the CA_RECLAIM in the parmlib and activated it as described previously, reorganize your DFSMSrmm control data set a final time. To check your DFSMSrmm control data set, run an IDCAMS EXAMINE as shown in Figure 3-35 to see if there is a need to reorganize your DFSMSrmm control data set.

```
//STEP1  EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//SYSIN    DD    *
  EXAMINE  -
       NAME(RMM.CONTROL.DSET) -
       INDEXTEST         -
       DATATEST
/*
```

Figure 3-35 Sample JCL to run IDCAMS EXAMINE

Check message IDC01728I (see Figure 3-36 on page 112). When there are empty control areas left, reorganize your DFSMSrmm control data set.

```
  EXAMINE  -
       NAME(RMM.CONTROL.DSET) -
       INDEXTEST         -
       DATATEST
IDC01700I INDEXTEST BEGINS
IDC11773I             133 KEYS PROCESSED ON INDEX LEVEL   1, AVERAGE KEY LENGTH:
42.4
IDC11773I             117 KEYS PROCESSED ON INDEX LEVEL   2, AVERAGE KEY LENGTH:
22.7
IDC11773I               2 KEYS PROCESSED ON INDEX LEVEL   3, AVERAGE KEY LENGTH:
25.5
IDC11774I CURRENT INDEX CISIZE IS  2048, RECOMMENDED MINIMUM INDEX CISIZE IS  1536
IDC01724I INDEXTEST COMPLETE - NO ERRORS DETECTED
IDC01701I DATATEST BEGINS
IDC01728I FOUND 113 EMPTY CONTROL AREAS THAT HAVE NOT BEEN RECLAIMED
IDC01709I DATATEST COMPLETE - NO ERRORS DETECTED
IDC01708I 133 CONTROL INTERVALS ENCOUNTERED
IDC01710I DATA COMPONENT CONTAINS 524 RECORDS
IDC01711I DATA COMPONENT CONTAINS 113 DELETED CONTROL INTERVALS
IDC01712I MAXIMUM LENGTH DATA RECORD CONTAINS 1036 BYTES
IDC01722I 99 PERCENT FREE SPACE
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

Figure 3-36 Sample EXAMINE output

ALLOCxx (optional): Allocation system default

Update the storage management subsystem definition member to specify the policy about whether to recall a migrated data set when you use an IEFBR14 JCL program with DD DISP=(x,DELETE) to delete the data set. Figure 3-37 shows the correct use of the IFBR14_DELMIGDS command.

This command specifies the policy about whether to recall a migrated data set when you use an IEFBR14 JCL program with DD DISP=(x,DELETE) to delete the data set. The recall is unnecessary in most cases because the data set is being deleted.

```
SYSTEM IEFBR14_DELMIGDS(NORECALL)
```

Figure 3-37 Sample IFBR14_DELMIGDS command

The following values apply:

LEGACY Indicates that the system is to recall hierarchical storage management (HSM)-migrated data sets before deletion.

NORECALL Indicates that the system can delete (through HSM HDELETE processing) the data set without first recalling the data set to the primary storage.

The default is LEGACY.

Note: This new function is available in z/OS V1.12 or higher.

AUTORxx (optional): Auto-reply policy specifications

Use the parmlib member AUTORxx to activate auto-reply processing on a system. The member AUTOR00 contains the auto-reply policy that is suggested by IBM. You can modify the member (which is not advised), or define another AUTORxx member to customize the auto-reply policy. The member AUTOR00 also contains comments that provide the message text for each WTOR and the rule that is used to select the WTOR.

Syntax rules for AUTORxx

The following syntax rules apply to the AUTORxx parmlib member:

- Data is specified in columns 1-71.
- Comments can start in any column and can span lines, and must start with /* and end with */. Comments might appear in any place where a blank is accepted, except within quoted strings.
- Generally, syntax errors cause the auto-reply changes to be rejected, while in the following cases, syntax errors are ignored and the changes are allowed to become active:
  - The maximum number of message IDs is reached.
  - Duplicate message IDs are specified.
  - NOTIFYMSGS is specified in different members.

Syntax format of AUTORxx

Use the syntax as shown in Figure 3-38 to add your DFSMSrmm-related messages.

```
                 |—HC————|
▸▸— NOTIFYMSGS(————————)————————————————————————————————————◂◂
                 |       |
                 |—CONSOLE—|

                      |—NOAUTOREPLY————————————————————————————|
                      |                                        |
                      |            |—m—|          |——,————————|  |
                      |            |—s—|          ▼             |  |
                      | DELAY(—nnn———)———REPLY(——['']replytext['']——)—|
▸▸— MSGID('msgid')————————————————————————————————————————————◂◂
```

Figure 3-38 AUTORxx command syntax

Note: The maximum number of message IDs that do not contain wildcards is limited to 10,413. The maximum number of message IDs that contain wildcards is limited to 1,500.

Updating AUTO-reply policy and WTOR information

You can use the MVS SET AUTOR command that is shown in Figure 3-39 to change the current AUTO-reply policy information.

```
SET AUTOR=(RM,00)

or

SET AUTOR=00
```

Figure 3-39 Update AUTO-reply and WTOR information

The two alphanumeric characters indicate the AUTORxx members of the logical PARMLIB that contain the specified AUTO-reply policy information. AUTOR allows you to change the current AUTO-reply policy specified in one or more AUTORxx parmlib members. Do not use parentheses when only one parmlib member is specified.

After the new setting is in effect, you get the information shown in Figure 3-40.

```
SET AUTOR=(RM,00)
IEE252I MEMBER  AUTORRM FOUND IN SYS1.IBM.PARMLIB
IEE252I MEMBER  AUTOR00 FOUND IN SYS1.PARMLIB
CNZ2600I AUTO-REPLY POLICY MODIFIED.
```

Figure 3-40 Display AUTOR command output

Note: If you specify only one parmlib member with APPC=, ASCH=, AUTOR=, CEE=, CNGRP=, GRSRNL=, MPF=, OMVS=, PROG=, or SCH=, you do not need to enter the parentheses.

Displaying auto-reply policy and WTOR information

You can use the DISPLAY AUTOR command that is shown in Figure 3-41 to display the current auto-reply policy.

```
DISPLAY AUTOR,POLICY
```

Figure 3-41 Display your current AUTOR settings

Example 3-10 contains the result of a DISPLAY AUTOR,POLICY command.

Example 3-10 DISPLAY AUTOR,POLICY command output

```
RESPONSE=SC70
 CNZ2603I 16.41.49 AUTOR POLICY 217
 POLICY ACTIVATED AT 16.28.35 ON 10/01/2010  NOTIFYMSGS(HC)
 FROM PARMLIB MEMBERS RM,00
 --MSG ID-- DELAY MEM ----REPLY TEXT----
...
 EDG0103D     1M   RM RETRY
 EDG0107A     1M   RM 00
 EDG0115D     1M   RM CANCEL
 EDG0117D     1M   RM CANCEL
 EDG0123D     5M   RM N
 EDG0127D     5M   RM CANCEL
 EDG0215D     2M   RM N
 EDG0228D     2M   RM CANCEL
 EDG0358D    10M   RM CANCEL
 EDG0361D    10M   RM RETRY
 EDG1107D     2M   RM RESTART
 EDG1200D    30S   RM CANCEL
 EDG1203D    30S   RM CANCEL
 EDG2103D    30S   RM L
 EDG2106D    30S   RM L
 EDG3213D    30S   RM RETRY
 EDG4000D     2M   RM RETRY
 EDG4001D     2M   RM RETRY
 EDG4010D     1M   RM RETRY
 EDG4012D     1M   RM RETRY
```

```
   EDG6605D    10M   RM RETRY
   EDG6607D    10M   RM RETRY
   EDG6609D    10M   RM RETRY
   EDG6626A    10M   RM END
   EDG6627A    10M   RM S
   EDG6629D     1M   RM CANCEL
   EDG6663D    10M   RM F
   EDG6671D    10M   RM F
   EDG8008D     1M   RM RETRY
   EDG8010D     1M   RM RETRY
   EDG8011D     1M   RM RETRY
   EDG8013D     1M   RM RETRY
   EDG8102D     1M   RM RETRY
   EDG8108D     1M   RM RETRY
   EDG8110D     1M   RM RETRY
   EDG8113D     1M   RM RETRY
   EDG8121D     1M   RM RETRY
   EDG8122D     1M   RM RETRY
   EDG8123D     1M   RM RETRY
  ....
```

The messages that are shown in Example 3-10 on page 114 are the default messages that are defined in the AUTORRM parmlib member. These messages can be used if DFSMSrmm is running in protect mode.

The following values apply:

column 1 The messageID in the range of 1 - 10 characters

column 2 The amount of time, in minutes (M) or seconds (S), in the range between 0 - 999, to wait, after the WTOR is issued and before auto-reply processing replies to the WTOR

column 3 The AUTORxx parmlib member suffixes that specified the policy

column 4 The reply that is used if auto-reply processing replies to the WTOR

You can use the DISPLAY AUTOR command that is shown in Figure 3-42 to display the current WTORs that are being monitored by auto-reply processing.

```
DISPLAY AUTOR,WTOR
```

Figure 3-42 Display your current AUTOR processing

Example 3-11 contains the result of a DISPLAY AUTOR,WTOR command.

Example 3-11 DISPLAY AUTOR,WTOR command output with pending actions

```
D AUTOR,W
CNZ2604I 06.47.02 AUTOR WTORS
0009 STATUS=06.47.58 SYS=SY1
MSG=IEE800D CONFIRM VARY FORCE FOR 3D0 - REPLY NO OR YES
REPLY=NO
0008 STATUS=06.47.46 SYS=SY1
MSG=IXC371D CONFIRM REQUEST TO VARY SYSTEM SY1 OFFLINE. REPLY
SYSNAME=SY1 TO REMOVE SY1 OR C TO CANCEL.
REPLY=C
```

Example 3-12 contains the result of a DISPLAY AUTOR,WTOR command if there is no action pending.

Example 3-12 DISPLAY AUTOR,WTOR command output without pending actions

```
RESPONSE=SC70
 CNZ2604I 16.50.20 AUTOR WTORS 220
 NO PENDING AUTO-REPLIES TO WTORS
```

Note: This new function is available in z/OS V1.12 or higher.

3.1.5 Enabling ISPF Data Set List (DSLIST) support

To enable direct entry into the DFSMSrmm ISPF dialog from the ISPF Data Set List Utility, use the ISPF Configuration Utility to update the ISPF configuration table. To enable this function, select the **Enable RM/Tape Commands** option. For information about how to use the ISPF Configuration Utility, see *z/OS ISPF Planning and Customizing,* GC34-4814. See Figure 16 on page 50 of *z/OS ISPF Planning and Customizing,* GC34-4814, which shows that the Data Set List Utility support is enabled, and shows the default values for the RM/Tape Command EDGRPD34 and Command APPLID EDG. You do not need to change these values.

Implementation steps

The ISPF Data Set List support can be implemented for a single user or it can be an installation-wide setting. Follow the steps to implement ISPF Data Set List support for a single user:

1. Create a new partitioned data set (PDS) with VB 255 to store the ISPF default settings.
2. Under ISPF, issue `TSO ISPCCONF` on any command line.
3. Select Option 1 Create/Modify Settings and Regenerate Keyword File on the ISPF Configuration Utility primary panel and add the name to the keyword file dsname field that you have allocated before. Then, add a member name. You get a message "`Keyword file loaded`."
4. Select Option 3 PDF Exits and Other PDF Settings on the ISPF Configuration Utility primary panel.
5. Scroll to the correct place and add / to Enable RM/Tape Commands.
6. Exit the dialog by pressing PF3 twice. You see the message "`Keyword file saved`."
7. Select Option 4 Build Configuration Table Load Module.
8. Specify any library in your ISPLLIB concatenation as the Output Configuration Table Load Module Data Set.
9. Press Enter to load the module that is built.
10. Exit ISPF and ensure that the specified load library is ISPLLIB-concatenated.
11. Start ISPF.

To implement the ISPF Data Set List support for all users, you can follow the steps as described previously, but the configuration table load module must be stored in the SISPLPA library. To load the module to the LPA, you must IPL your system. For detailed information, see 10.4, "Move the ISPCFIGU module to the SISPLPA library" on page 367.

For more information, see Chapter 10, "Enabling ISPF Data Set List (DSLIST) support" on page 357.

3.1.6 REXX Variable Constraint Relief

A new operand, VARSTORAGE, was added to the PROFILE command to specify whether variables in the CLIST or authorized REXX variable pools can use storage above the 16 MB line.

If you have a large environment and you need to list a large number of volumes normally, you always get a return code 4 reason code 10 "`Insufficient storage for search processing`." The message "`More records might exist`" is avoided.

If your TSO LOGON session parameter size is 30000, approximately 2,400 volumes can be retrieved with the setting VARSTORAGE(LOW). If you change your settings to VARSTORAGE(HIGH), you can retrieve approximately 20,000 volumes.

VARSTORAGE specifies the storage location to be used for CLIST variables or REXX OUTTRAP variables that contain output from authorized commands. A CLIST or REXX exec uses the VARSTORAGE setting of the PROFILE command when it starts. This setting then remains unchanged for the life of the CLIST or REXX exec, even if the CLIST or REXX exec issues a new PROFILE command with a different VARSTORAGE setting. The new setting will only apply when a new CLIST or REXX exec begins.

VARSTORAGE (HIGH)
Indicates that CLIST variables and REXX OUTTRAP variables that contain output from authorized commands that are invoked by REXX can be kept in storage above the 16 MB line.

VARSTORAGE (LOW)
Indicates that CLIST variables and REXX OUTTRAP variables that contain output from authorized commands that are invoked by REXX can only be kept in storage below the 16 MB line. If you specify VARSTORAGE with no operands, VARSTORAGE(LOW) is the default. This is the default value when your user profile is created.

To show your current settings, use the TSO PROFILE command as shown in Figure 3-43 without any additional operands.

```
TSO PROF
```

Figure 3-43 Display your current profile settings

Figure 3-44 shows your current profile settings, including the new VARSTORAGE information. If you have not set the VARSTORAGE information, you can see that the default setting LOW is set.

```
CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE NOMSGID MODE    WTPMSG NORECOER
PREFIX(MHLRES5) PLANGUAGE(ENU) SLANGUAGE(ENU) VARSTORAGE(LOW) DEFAULT
LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
```

Figure 3-44 Your current profile settings

If you have requested to search for all volumes in your DFSMSrmm database, you normally see the message "`More volumes may exist`" on the panel as shown on Figure 3-45 on page 118 and that the search ended before all volumes were listed.

```
        Panel  Help  Scroll
 ____________________________________________________________________________
                    DFSMSrmm Volumes (Page 1 of 2)       More volumes may exist
 Command ===>                                                Scroll ===> CSR
 There is not enough storage available to list all the volumes
 Enter HELP or PF1 for the list of available line commands
 Use the RIGHT command to view other data columns
    Volume          Assigned   Expiration                  Dest-    Tra- Data
 S  serial Owner    date       date       Status  Location ination  nsit sets
 -- ------ -------- ---------- ---------- ------- -------- -------- ---- -----
    AFS201 VGRMMLIB 2004/303   1999/365   USER    SHELF             N    0
    AFS202 VGRMMLIB 2004/303   1999/365   USER    SHELF             N    0
    BTS001 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS005 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS006 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS007 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS008 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS010 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS011 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
    BTS012 VGRMMLIB 2005/005   1999/365   USER    SHELF             N    0
```

Figure 3-45 Display incomplete search result

To update the VARSTORAGE value, use the TSO PROFILE command as shown in Figure 3-46.

```
TSO PROFILE VARSTORAGE(HIGH)
```

Figure 3-46 Update your TSO PROFILE settings

Figure 3-47 shows that the new profile settings include the new VARSTORAGE information.

```
CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE NOMSGID MODE    WTPMSG NORECOER
PREFIX(MHLRES5) PLANGUAGE(ENU) SLANGUAGE(ENU) VARSTORAGE(HIGH) DEFAULT
LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
```

Figure 3-47 TSO profile settings

3.1.7 Creating a starting procedure

This section describes the system symbols and the DFSMSrmm problem determination aid.

Displaying your system symbols

System symbols are elements that allow systems to share parmlib definitions while retaining unique values in those definitions. You can code the statements in the IEASYMxx member of parmlib, so only one IEASYMxx member is needed to define static system symbols and system parameters for all systems in a multisystem environment. You can code additional IEASYMxx members to organize the definitions more clearly.

If you have to define DFSMSrmm on multiple systems, you can check the symbols that are defined in your system before you create the procedure. To check all the defined symbols in an IEASYMxx member, you can use the following MVS DISPLAY SYMBOLS command that is shown in Figure 3-48 to get a list of all symbols and their values.

```
DISPLAY SYMBOLS
```

Figure 3-48 MVS DISPLAY SYMBOLS command syntax

Example 3-7 shows all the available system-wide symbols and the values on your system.

Example 3-13 Display SYMBOLS output

```
DISPLAY SYMBOLS
IEA007I STATIC SYSTEM SYMBOL VALUES 715
        &SYSALVL.  = "2"
        &SYSCLONE. = "70"
        &SYSNAME.  = "SC70"
        &SYSPLEX.  = "SANDBOX"
        &SYSR1.    = "Z1DRB1"
        &ALLCLST1. = "CANCEL"
        &CMDLIST1. = "70,00"
        &COMMDSN1. = "COMMON"
        ...
        &SECSUBSY. = "JES3"
        &SMFPARMS. = "00"
        &SRES2.    = "Z1DRB2"
        &SYSLEVEL. = "Z0SR1D"
        &SYSR2.    = "Z1DRB2"
        &SYSR3.    = "Z1DRB3"
        &VOL01.    = "MLDC65"
```

Using the Problem Determination Aid facility

To analyze problems when running DFSMSrmm, you can use the Problem Determination Aid (PDA) facility to gather related information. The PDA facility is required for IBM service because it traces module and resource flow. The PDA facility consists of in-storage trace, optional DASD log data sets, EDGRMMxx parmlib member options, and operator commands to control tracing.

If you need an external DASD record of trace data, perform this step once for each MVS image.

Two separate log data sets are used by the PDA facility. DFSMSrmm recognizes these log data sets by using EDGPDOX and EDGPDOY DD statements. The EDGPDOX and EDGPDOY data sets must be pre-allocated and cataloged to be used by DFSMSrmm. Both data sets must be allocated on the same volume. The DFSMSrmm RACF user ID requires ALTER access to the EDGPDOX and EDGPDOY data sets.

Before you can use the PDA facility, you need to perform these tasks:

- Determine how long you want to keep trace information
- Optionally allocate storage on DASD for the PDA log data sets: EDGPDOX and EDGPDOY
- Implement the PDA facility based on how long you want to keep trace data

For detailed information about the PDA facility, see *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Updating the procedure library

Create a procedure in SYS1.PROCLIB to start the DFSMSrmm subsystem address space. Example 3-8 shows the sample JCL.

```
//DFRMM   PROC M=00,OPT=MAIN
//IEFPROC EXEC PGM=EDG&OPT.,PARM='&M',TIME=1440,REGION=0M
//PARMLIB DD   DDNAME=IEFRDER
//IEFRDER DD   DISP=SHR,DSN=SYS1.PARMLIB
//EDGPDOX DD   DISP=SHR,DSN=RMM.&SYSNAME..RMMPDOX
//EDGPDOY DD   DISP=SHR,DSN=RMM.&SYSNAME..RMMPDOY
```

Figure 3-49 Sample DFSMSrmm starting procedure

In this example, we use REGION=0M to get the maximum region size. If your default region size is small, we suggest that you specify REGION=40M or more.

The following values apply to Figure 3-49:

DFRMM Specifies the procedure name. You can use any procedure name from 1 - 8 characters long.

M Allows you to specify a parmlib member suffix. M=00 tells DFSMSrmm to use EDGRMM00.

OPT Allows you to specify one of these options:

–OPT=RESET to disable the subsystem interface
–OPT=MAIN to execute the DFSMSrmm subsystem

> **Restriction:** Before DFSMSrmm disables the subsystem interface, it ensures that the user who made the request is authorized. In this case, the RACF user of the STARTED class (STC) needs ALTER access to the STGADMIN.EDG.RESET.SSI resource defined in RACF FACILITY class.

EDGPDOX An optional statement that is required only for an external trace output recording. If not specified, no logging of the PDA trace output is performed. The data set specified must be pre-allocated and cataloged. The DFSMSrmm RACF user ID requires ALTER authority to this data set. This must be allocated on the same volume as EDGPDOY.

EDGPDOY An optional statement that is required only for external trace output recording. The same considerations for EDGPDOX apply to EDGPDOY.

> **Important:** You must define a pair of trace output data sets for each MVS image. Do not share the trace data sets with multiple DFSMSrmm systems or with other system components.

&SYSNAME This specifies a system symbol to use the same procedure on different systems.

REGION As you determine the REGION size for the DFSMSrmm started procedure, the amount of virtual storage that DFSMSrmm uses depends on the resources you have defined. DFSMSrmm virtual

storage usage can be affected by any REGION size controls or restrictions that your systems might have in place, such as in IEFUSI. The sample DFRMM procedure specifies REGION=60M, which normally provides all the private region below 16 MB and 60 MB above 16 MB. To enable DFSMSrmm to use all available virtual storage, specify REGION=0M. If you want to set a specific region size, consider these tips, along with the current region size of your DFRMM started procedure, to determine whether you need to make any changes to the REGION size.

For information about special considerations or restrictions when coding the starting procedure, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.1.8 Defining an alias for high-level qualifier RMM

We suggest that you create an alias in the integrated catalog facility (ICF) user catalog for RMM as shown in Figure 3-50.

```
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
   DEFINE ALIAS                    -
          (NAME(RMM)               -
          RELATE('SYSCAT1.SYST01'))
```

Figure 3-50 Defining an alias in an ICF user catalog

The DEFINE ALIAS command defines an alias, RMM, for the user catalog SYSCAT1.SYST01. All catalog entries for data sets defined with a high-level qualifier (HLQ) of RMM are defined in user catalog SYSCAT1.SYST01.

3.1.9 Protecting DFSMSrmm resources

To protect DFSMSrmm functions, you need to use an external security product, such as RACF. Invocation to this product needs to be through the security authorization facility (SAF). If RACF is not installed, you must provide equivalent function through the SAF interface.

The security implementation tasks are listed:

1. Assign a RACF user ID to DFSMSrmm.
2. Identify DFSMSrmm to RACF.
3. Define RACF groups for DFSMSrmm.
4. Define DFSMSrmm resources to RACF.
5. Check the STGADMIN.ADR.DUMP.CNCURRNT resource.

Before you start DFSMSrmm, you need to create a RACF user ID for DFSMSrmm and define the DFSMSrmm started task to RACF. You can find more detailed information about how to implement RACF security for DFSMSrmm, and sample definitions, in Appendix A, “Security topics” on page 633.

Assigning a RACF user ID to DFSMSrmm

You can assign a RACF user ID that matches the name of the DFSMSrmm procedure you created in 3.1.7, “Creating a starting procedure” on page 118, but any installation-selected RACF user ID is acceptable. Because data sets are created for use by the DFSMSrmm procedure, add the DFSMSrmm RACF user ID to the access list for the data sets.

We used the RACF command that is shown in Figure 3-51 to define the DFSMSrmm user ID.

```
ADDUSER DFRMM DFLTGRP(SYS1) NAME('DFSMSrmm Userid')
```

Figure 3-51 Define a new RACF user

Note: If you do not specify the DFLTGRP parameter on the ADDUSER command, the current connect group of the user issuing the ADDUSER command is used as the default.

Identifying DFSMSrmm procedures to RACF

To assign RACF identities to started procedures, you can use the RACF STARTED class to add or modify security definitions for new and existing started procedures.

Enter the RACF commands that are described in Figure 3-52 to assign RACF identities to the DFSMSrmm started procedure.

```
RDEFINE STARTED (DFRMM.*) UACC(NONE) STDATA(USER(DFRMM) GROUP(SYS1))
SETROPTS RACLIST(STARTED) REFRESH
SETROPTS GENERIC(STARTED) REFRESH
```

Figure 3-52 Assigning RACF identities to the DFSMSrmm started procedures

Note: Remember to refresh the in-storage profiles, using the SETROPTS REFRESH command, after you add profiles to the STARTED class. If you plan to use the EDGLABEL, EDGXPROC, or the EDGBKUP procedures, you must define the procedures in the RACF STARTED class. Before RACF 2.1, no STARTED class was available for use. The only way to associate a started procedure with a RACF user ID was by coding the RACF started procedures table, ICHRIN03.

3.1.10 Defining RACF resources and groups for DFSMSrmm

To implement RACF security for DFSMSrmm resources, define a set of user groups to handle different kinds of resources at different access levels. For more information, see Appendix A, “Security topics” on page 633.

STGADMIN.EDG.RESET.SSI FACILITY class profile

At this stage, there is one security profile in particular that you need to be aware of and create. When you first install and start to use DFSMSrmm, you likely need to stop or even remove DFSMSrmm from your system. Although you can easily stop DFSMSrmm, if you have EDGSSSI in the subsystem name table DFRM entry, or have been using a running mode that supports tape recording, DFSMSrmm prevents all tape usage until it is restarted. To allow tapes to be used, you must restart DFSMSrmm or remove it from the system.

The RACF FACILITY class profile STGADMIN.EDG.RESET.SSI controls the use of the RESET facility for removing DFSMSrmm from the system. Be sure to define this security profile and authorize the DFSMSrmm started task to reset the subsystem interface. Grant the DFSMSrmm started task procedure RACF user ID ALTER access authority to STGADMIN.EDG.RESET.SSI. For specific details about removing DFSMSrmm from the system, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Note: You can use the RESET facility without defining this profile when you do not have a security product installed.

3.1.11 Creating the DFSMSrmm CDS

The DFSMSrmm control data set (CDS) is a VSAM key-sequenced data set (KSDS) that contains the complete inventory of the removable media library. DFSMSrmm records all changes that are made to the inventory, such as adding or deleting volumes in the CDS.

You can define the DFSMSrmm control data set as either an extended format (EF) or a non-extended format VSAM data set. Using a non-extended format data set for the DFSMSrmm control data set limits the control data set size to a maximum of 4 GB. Using an EF data set enables you to use VSAM functions, such as multivolume allocation, compression, or striping. EF also enables you to define a control data set that uses VSAM extended addressability (EA) to enable the control data set to grow above 4 GB.

To define an EF control data set, you must include the DATACLASS keyword on the access method services (AMS) DEFINE command and reference the correct data class. For more information about EF data sets, see *z/OS DFSMS: Using Data Sets,* SC26-7410. For information about defining data classes with DSNTYPE=EXT and EXTENDED ADDRESSABILITY=Y, see *DFSMSdfp Storage Administration Reference*, SC26-7402.

DFSMSrmm requires CONTROL access to the control data set. The control data set cannot be a SYS1.xx data set if the control data set is to be shared. Additionally, batch local shared resources (LSR) cannot be used with the DFSMSrmm control data set. In our example, we use the non-extended format.

Note: All data sets that are used by DFSMSrmm can be eligible for allocation in the extended addressing space of an extended address volume (EAV). This includes the DFSMSrmm journal and any dynamically allocated temporary files.

To create the CDS on the system, use the following steps:

1. Calculate the space for the DFSMSrmm CDS.
2. Protect the CDS.
3. Allocate the DFSMSrmm CDS.
4. Initialize the CDS using the EDGJUTIL utility.

Calculating the space for the DFSMSrmm CDS

There is sample JCL in SAMPLIB member EDGJMFAL to allocate the CDS. You can modify it to meet your installation standards.

The control data set is one of the most important components of DFSMSrmm. Therefore, to avoid space problems while running DFSMSrmm, you must calculate the space required before its allocation.

Table 3-5 shows the space requirement for each kind of record in the CDS.

Table 3-5 DFSMSrmm CDS space requirement

Control data set content	DASD space
Control record	1 MB (MB equals approximately 1,000,000 bytes)
Data sets	584 KB for every 1,000 data sets
Shelf locations in the library that do not contain volumes	164 KB for every 1,000 shelf locations
Shelf locations in storage locations	140 KB for every 1,000 shelf locations
Owners	38 KB per 1,000 volumes
Software products, average five volumes per product	420 KB for every 1,000 software products
Volumes	1 MB for every 1,000 volumes
VRSs	212 KB for every 1,000 VRSs

Protecting the CDS

To protect the DFSMSrmm CDS, define a discrete profile, and then permit the appropriate RACF groups to the RACF data set profile. Figure 3-53 contains sample JCL. For detailed information about implementing RACF security for DFSMSrmm, see Appendix A, “Security topics” on page 633.

```
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
   ADDSD 'RMM.PROD.CDS' UACC(NONE) GENERIC
      PERMIT 'RMM.PROD.CDS' GENERIC-
             ID(EDGADMIN) ACC(READ)
      PERMIT 'RMM.PROD.CDS' GENERIC-
             ID(EDGLIB) ACC(READ)
      PERMIT 'RMM.PROD.CDS' GENERIC-
             ID(EDGSYSPG) ACC(ALTER)
```

Figure 3-53 Sample JCL to protect the DFSMSrmm CDS

Allocating the DFSMSrmm CDS

The JCL to create the DFSMSrmm CDS in Figure 3-54 on page 125 has been modified to use special attributes.

```
//STEP02   EXEC PGM=IDCAMS ALLOCATION MASTER FILE
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,UNIT=3390,VOL=SER=DFRMM4
//SYSIN    DD   *
   DEFINE CLUSTER(NAME(RMM.PROD.CDS)     -
          FILE(MASTER)                   -
          FREESPACE(15 0)                -
          KEYS(56 0)                     -
          REUSE                          -
          RECSZ(512 9216)                -
          SHR(3 3)                       -
          KILOBYTES(4500 1500)           -
          STORAGECLASS(gspace)           -
          VOLUMES(DFRMM4))               -
   DATA(NAME(RMM.PROD.CDS.DATA)          -
          BUFFERSPACE(829440)            -
          CISZ(26624))                   -
   INDEX(NAME(RMM.PROD.CDS.INDEX)        -
          CISZ(2048))
/*
```

Figure 3-54 Sample JCL to allocate the DFSMSrmm CDS

Note: For initial testing with DFSMSrmm, a small CDS is sufficient. When you convert your existing data to DFSMSrmm, you need a CDS that is sized to accommodate your existing and planned requirements.

Initializing the CDS using the EDGJUTIL utility

Before using the DFSMSrmm CDS, a control record must be written in the CDS using the EDGUTIL utility. The control record contains information about the number of shelf locations in the library and storage locations.

Sample JCL is provided in SYS1.SAMPLIB member EDGJUTIL, which you can modify to meet your installation standards. Figure 3-55 shows a sample JCL that can be used to initialize the CDS.

```
//EDGUTIL  EXEC PGM=EDGUTIL,PARM='CREATE'
//SYSPRINT DD   SYSOUT=*
//MESSAGE  DD   SYSOUT=*
//MASTER   DD   DISP=OLD,DSN=RMM.PROD.CDS
//SYSIN    DD   *
  CONTROL CDSID(PROD) EXTENDEDBIN(YES) CATSYNCH(NO)
/*
```

Figure 3-55 Sample JCL to initialize the DFSMSrmm CDS

The following values are shown in Figure 3-55:

CDSID(PROD) Specifies 1 - 8 characters that identify the control data set by name.

CATSYNCH(NO) Specify CATSYNCH(NO) to force synchronization of the DFSMSrmm control data set and user catalogs the next time inventory management is run. For more information, see Chapter 9, “Catalog synchronization” on page 329.

EXTENDEDBIN(YES)

Enables DFSMSrmm extended bin support, which allows the reuse of bins at the start of a move.

Extended bin support must be enabled if you want to use the DFSMSrmm PARMLIB OPTION command REUSEBIN(STARTMOVE) operand as shown in Example 3-5 on page 96 to reuse bins when a volume moves from a bin.

3.1.12 Creating the DFSMSrmm CDS as extended format

You can define the DFSMSrmm CDS as an extended format (EF) VSAM data set. This function is enabled with APAR OW47639. Extended format data sets offer these benefits:

- Data striping
- Data compression
- VSAM extended addressability
- Partial space release
- System-managed buffering
- CDS above 4 GB

The extended format enables you to define a CDS that uses VSAM Extended Addressability (EA) to enable the CDS to grow above 4 GB.

Extended format data sets must be system-managed. The mechanism to request extended format is through the SMS data class attribute Data Set Name Type, specifying a value of `EXT`. For an extended format data set to be allocated using extended addressability, the attribute Extended Addressability must be set to `Y` (see Figure 3-56).

```
                              DATA CLASS DEFINE                Page 2 of 4
Command ===>

SCDS Name . . . : SYS1.SMS.SCDS
Data Class Name : EFKSDS

To DEFINE Data Class, Specify:

  Data Set Name Type  . . . . . . EXT  (EXT, HFS, LIB, PDS or blank)
    If Ext  . . . . . . . . . . . R    (P=Preferred, R=Required or blank)
    Extended Addressability . . . Y    (Y or N)
    Record Access Bias  . . . . .      (S=System, U=User or blank)
  Space Constraint Relief . . . . N    (Y or N)
    Reduce Space Up To (%)  . . .      (0 to 99 or blank)
    Dynamic Volume Count  . . . .      (1 to 59 or blank)
  Compaction  . . . . . . . . . .      (Y, N, T, G or blank)
  Spanned / Nonspanned  . . . . .      (S=Spanned, N=Nonspanned or blank)
```

Figure 3-56 Data class specifying extended format

To define an extended format or extended addressability DFSMSrmm CDS, you must include the DATACLAS keyword as shown in Figure 3-57 on page 127, and reference the correct data class. For more information about how to reference a data class, see *DFSMS Access Method Services for Catalogs*, SC26-7394.

```
//STEP01   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,UNIT=3390,VOL=SER=DFRMM4
//SYSIN    DD   *
   DEFINE CLUSTER(NAME(RMM.PROD.CDS)     -
          FILE(MASTER)                   -
          FREESPACE(15 0)                -
          KEYS(56 0)                     -
          REUSE                          -
          RECSZ(512 9216)                -
          SHR(3 3)                       -
          KILOBYTES(4500 1500)           -
          STORAGECLASS(gspace)           -
          DATACLASS(EFKSDS)              -
          VOLUMES(DFRMM4))               -
   DATA(NAME(RMM.PROD.CDS.DATA)          -
          BUFFERSPACE(829440)            -
          CISZ(26624))                   -
   INDEX(NAME(RMM.PROD.CDS.INDEX)        -
          CISZ(2048))
/*
```

Figure 3-57 Sample JCL to allocate an extended format DFSMSrmm CDS

3.1.13 Creating the DFSMSrmm journal

The DFSMSrmm journal contains a record of all changes that are made to the CDS since the last backup. We suggest that you create the journal and use it to forward recover changes that are made since the last backup. Each time you successfully back up the CDS by using the EDGHSKP utility, the journal data set is cleared.

We suggest that you allocate the journal data set on a different volume than the CDS, back up the journal data set, and maintain multiple generations of it.

To create the DFSMSrmm journal on the system, perform the following steps:

1. Calculate the space of the DFSMSrmm journal.
2. Protect the DFSMSrmm journal.
3. Allocate the DFSMSrmm journal.

Calculating the space of the DFSMSrmm journal

Table 3-6 shows the space requirement for each kind of record in the journal data set.

Table 3-6 DFSMSrmm journal space requirement

Journal content	DASD space
Changes by users	1.5 KB for each change made
Data sets	1.5 KB for each data set retained by a VRS
Data sets no longer retained by a VRS	1.5 KB for each data set no longer retained by a VRS
Expiring volumes 1.5 KB for each expiring volume	1.5 KB for each expiring volume
Non-scratch mounts	6.7 KB for each mount
Scratch mounts	8.3 KB for each mount

Journal content	DASD space
Volumes	1.5 KB for each volume that is retained by a VRS
Volume checked in/out	2.6 KB for each volume in or out of the library
Volumes returned to scratch	3 KB for each volume returned to scratch
Volumes to and from storage locations	3.5 KB for each volume moved to or from a storage location
Volumes no longer retained by a VRS	1.5 KB for each volume that has not reached its expiration date and is no longer retained by a VRS
Volumes that are exported or imported	1.5 KB for each logical volume that is exported or imported
VRSs	1.3 KB for each VRS that is created

Divide the total KB of space by 4, because allocation is by average record size using a 4 K value. Also, use this number in the space allocation (SPACE=) that is shown in Figure 3-59 on page 129.

Important: If you have specified JRNLTRAN(YES), plan to provide up to 33% more journal data set space to accommodate the additional records.

Protecting the DFSMSrmm journal

To protect the DFSMSrmm journal, define a discrete profile and then permit the appropriate RACF groups to the RACF data set profile. Figure 3-58 contains sample JCL. For detailed information about implementing RACF security for DFSMSrmm, see Appendix A, “Security topics” on page 633.

```
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
   ADDSD 'RMM.PROD.JRNL' UACC(NONE) GENERIC
      PERMIT 'RMM.PROD.JRNL' GENERIC-
             ID(EDGADMIN) ACC(READ)
      PERMIT 'RMM.PROD.JRNL' GENERIC-
             ID(EDGLIB) ACC(READ)
      PERMIT 'RMM.PROD.JRNL' GENERIC-
             ID(EDGSYSPG) ACC(ALTER)
```

Figure 3-58 Sample JCL to protect the DFSMSrmm journal

Allocating the DFSMSrmm journal

Sample JCL to create the DFSMSrmm journal is provided in SYS1.SAMPLIB member EDGJNLAL. You can modify the sample JCL to meet your installation standards. Figure 3-59 on page 129 shows sample JCL to create the DFSMSrmm journal.

```
//STEP02   EXEC PGM=IEFBR14
//SYSPRINT DD   SYSOUT=*
//JOURNAL  DD   DISP=(,CATLG,DELETE),
//         DSN=RMM.PROD.JRNL,
//         UNIT=3390,VOL=SER=DFRMM3,
//         AVGREC=U,SPACE=(4096,(6957),,CONTIG)
```

Figure 3-59 Sample JCL to allocate the DFSMSrmm journal

For initial testing with DFSMSrmm, either no journal or a small journal is sufficient. Later, when you use converted data, you need to use a journal that is large enough for your daily needs.

3.1.14 Restarting MVS with DFSMSrmm implemented

Important: You can schedule the IPL to implement the DFRM subsystem any time ahead of migration and parallel running, but you need to issue the following command as part of that IPL before tapes are used:

```
S DFRMM,OPT=RESET
```

You can use this command only if you defined the started task DFRMM as shown in Figure 3-49 on page 120, and you defined the resource STGADMIN.EDG.RESET.SSI in the RACF class FACILITY as explained in 3.1.10, “Defining RACF resources and groups for DFSMSrmm” on page 122. The RACF user ID of the started task needs ALTER access to this profile.

You are ready to start the system with DFSMSrmm implemented. You might be able to dynamically implement changes to the MVS system PARMLIB members or modified installation exits depending on your MVS level. You must perform these tasks:

- IPL with create link pack area (CLPA) to include new levels of the DFSMSrmm code that have LPALIB as the target library.
- IPL when you make changes to IFAPRDxx to change the licensing options.
- IPL to implement the DFRM subsystem if you plan to run DFSMSrmm in parallel with any other tape management system.

  You can implement DFSMSrmm without the need of an IPL if you use the TSO commands and MVS commands described previously.

  Important: If you installed a z/OS release before z/OS V1.3 and your tape management system is running, you cannot update the subsystem name table dynamically to implement DFSMSrmm. In this case, you have to re-IPL your MVS system with CLPA.

  If you installed DFSMSrmm Release 2.10, you can update the subsystem name table dynamically to implement DFSMSrmm, but there are error messages, such as EDG0003E or EDG0004E, which you can ignore. When you are prompted by the EDG0103D message, reply by entering `RETRY`.

With z/OS V1.3 and later releases, there is no longer a requirement to have DFSMSrmm started before other tape management systems. DFSMSrmm does not update the link pack area (LPA) or use dynamic hooks into DFSMSdfp. Therefore, the only need for an IPL with CLPA beginning with z/OS V1.3 is if any DFSMSrmm maintenance goes into LPALIB.

3.1.15 Starting and restarting DFSMSrmm

Usually, the DFSMSrmm subsystem starts automatically through standard initial program load (IPL). Under some conditions, you might need to start or restart the subsystem.

To start DFSMSrmm using the default procedure name and PARMLIB member, enter the S DFRMM MVS command shown in Figure 3-60.

```
S DFRMM
```

Figure 3-60 Start the DFRMM task

You can start DFSMSrmm with different parameters from the defaults. You can use the MVS command as shown in Figure 3-61 to specify another PARMLIB data set and member name.

```
S DFRMM,M=xx,DSN=parmlib_dataset
```

Figure 3-61 Start the DFRMM task with different parameters

The following information is shown in Figure 3-61:

M Specify the suffix of the EDGRMMxx member in PARMLIB.
DSN Specify a different PARMLIB data set name.

After this command, you will see the message shown in Figure 3-62.

```
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 3-62 EDG0105I initialization message

Figure 3-63 shows the message that tells you that the subsystem interface was not activated before.

```
EDG0103D DFSMSrmm SUBSYSTEM INTERFACE IS INACTIVE - ENTER "IGNORE",
             "CANCEL" OR "RETRY"
```

Figure 3-63 EDG0103D initialization message

Enter `RETRY` to continue with the initialization. After that, you receive the EDG0105I message.

You can restart DFSMSrmm using the MVS MODIFY command, for example:

- To restart DFSMSrmm and implement new PARMLIB options, specify the EDGRMM parmlib member suffix in the command as shown in Figure 3-64.

```
F DFRMM,M=xx
```

Figure 3-64 Restart DFRMM specifying a different member suffix

- To turn Problem Determination Aid (PDA) tracing on or off, you can use the commands that are shown in Figure 3-65.

```
F DFRMM,PDA=ON
F DFRMM,PDA=OFF
```

Figure 3-65 Turn problem determination on or off

- To manually quiesce DFSMSrmm to stop all tape-related activities and deallocate the control data set and journal, use the command that is shown in Figure 3-66.

```
F DFRMM,QUIESCE
```

Figure 3-66 Quiesce DFSMSrmm

- To refresh DFSMSrmm installation exits and enable exits in a z/OS environment earlier than V1R11 that are currently disabled, specify the command shown in Figure 3-67.

```
F DFRMM,REFRESH EXITS
```

Figure 3-67 Refresh installation exits prior z/OS V1R11

- With z/OS V1.11, the MODIFY DFRMM,REFRESH EXIT command is no longer supported. You can use the SETPROG EXIT command instead, which allows the exit modules to be managed independently. Also, remember that the EXIT statement of the PROGxx parmlib member can be used to add an exit module. In Figure 3-68, you can see how you can add, modify, or delete a single DFSMSrmm installation exit using the SETPROG operator command.

```
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300
SETPROG EXIT,MODIFY,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300,STATE=INACTIVE
SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300
```

Figure 3-68 SETPROG command syntax to manage exit modules

- To refresh an exit, you have to delete the exit first and then you must add the exit to reload the module from the library as shown in Figure 3-69.

```
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 3-69 Refresh installation exits with z/OS V1R11 or higher

Note: In addition to deleting and adding an exit module, you can also temporarily deactivate it. However, remember that setting the state to ACTIVE again does not imply a reload of the module from the library.

You can use the SETPROG EXIT command to control exits that have been defined to the dynamic exits facility. To activate or change the currently used EDGUX100 user exit, you have to use the command that is shown in Figure 3-70 on page 132. This replaces the old sequence that is used for releases earlier than z/OS V1.12.

```
SETPROG EXIT,REFRESH,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 3-70 SETPROG command syntax to refresh a dynamic exit

For more information about these commands, see *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.1.16 Stopping DFSMSrmm

The DFSMSrmm subsystem must always be activated in your system. In some recovery situations, you might need to remove DFSMSrmm from your system. In that case, you must follow this procedure:

1. All jobs that are processing and using tapes need to be completed. Before you stop DFSMSrmm, stop batch initiators to avoid failing a job that opens a tape data set. Also, obtain a list of the number of requests waiting to be processed and the number of active requests. Figure 3-71 shows the correct use of the query active command.

```
F DFRMM,QUERY ACTIVE
```

Figure 3-71 Query active command syntax

As a result of this command, you see the messages that are shown in Figure 3-72.

```
EDG1113I F=function requestor_type=requestor_name time TKN=token_value
EDG1114I COMMAND COMPLETE, TOTAL TASKS total_count, ACTIVE active_count
EDG1118I queued_count QUEUED REQUESTS, INCLUDING nowait_count NOWAIT
         catalog_count CATALOG
```

Figure 3-72 DFSMSrmm query active messages

The following values apply:

F Identifies the requested DFSMSrmm function; the values are internal to DFSMSrmm.

requestor_type Identifies the requester. It can be JOB, STC, or TSU.

time Lists the time that the request was started.

TKN Identifies the request.

total_count The number of tasks available to process DFSMSrmm requests.

active_count The number of tasks currently processing.

queued_count The number of requests that are waiting to be processed by DFSMSrmm.

nowait_count The number of requests that are waiting to be processed by DFSMSrmm for which the request does not wait for the results; includes the count of catalog requests.

catalog_count The number of requests that are waiting to be processed by DFSMSrmm to reflect catalog update activity in the DFSMSrmm control data set.

Figure 3-73 on page 133 shows sample output from this command. In this sample, the only request to DFSMSrmm is the HOUSEKEEP processing that is executed by the YCJRES1B job.

```
EDG1119I DFSMSrmm STATUS IS ACTIVE. JOURNAL ENABLED.
EDG1120I Function  System  Task   Name  Started   Token  S IP Status
EDG1113I HSKP              JOB=MHLRES7H 17:58:08 00300028      :  :
EDG1114I LOCAL TASKS  10, ACTIVE   1, SERVER TASKS   0, ACTIVE   0
EDG1122I                  HELD     0                    HELD     0
EDG1118I      0 QUEUED REQUESTS, INCLUDING      0 NOWAIT      0 CATALOG
EDG1121I DEBUG: DISABLED, PDA TRACE LEVEL: 1-2-3-4- RESERVE: 17:58:10
EDG1101I DFSMSrmm MODIFY COMMAND ACCEPTED
```

Figure 3-73 DFSMSrmm MODIFY command output

2. Optionally, remove the DFSMSrmm subsystem. Usually, you do not need to remove it. If you want to allow tape use without DFSMSrmm recording, you can use the reset command that is shown in Figure 3-74.

```
S DFRMM,OPT=RESET
```

Figure 3-74 Remove the DFSMSrmm subsystem

Important: To use this command, the user ID of the started task needs ALTER access to the resource STGADMIN.EDG.RESET.SSI that is defined in the RACF FACILITY class.

Wait for the response that is shown in Figure 3-75.

```
IEF403I DFRMM - STARTED - TIME=18.48.27 - ASID=0063.
EDG0181I DFSMSRMM SUBSYSTEM INTERFACE SUCCESSFULLY INACTIVATED
```

Figure 3-75 DFSMSrmm subsystem reset command output

The use of OPT=RESET must be RACF-protected. For more information about the RACF profiles, see “Defining DFSMSrmm resources to RACF” on page 636.

Important: The OPT=RESET operand must only be used during parallel running or testing, when DFSMSrmm must be completely removed from the system due to an error, or because you need to reconvert.

3. Shut down the DFSMSrmm subsystem with the command in Figure 3-76.

```
P DFRMM
```

Figure 3-76 Stop the DFRMM task

When you see the messages in Figure 3-77, DFSMSrmm is completely removed from your system.

```
IEF352I ADDRESS SPACE UNAVAILABLE
$HASP395 DFRMM    ENDED
```

Figure 3-77 Stop DFRMM messages

3.1.17 Setting up DFSMSrmm utilities

There are several DFSMSrmm utilities that you need to set up now:

- EDGHSKP is used to perform the following inventory management activities:
  - Vital record processing to determine which volumes to retain and which volume moves are required, based on VRSs
  - Expiration processing to identify volumes ready to be released and returned to scratch
  - Storage location management processing to assign shelf locations to volumes that are being moved to storage locations
  - Backup and recovery of the CDS and journal

    Use the DFSMSrmm backup utilities instead of other backup utilities, such as the IDCAMS EXPORT command, because DFSMSrmm provides the necessary serialization and forward recovery functions. DFSMSrmm backup utilities check whether the CDS is in use, tell the DFSMSrmm subsystem that backup or recovery is in process, and provide a way to forward recover.
  - Creation of an extract data set

    You can produce movement and inventory reports by producing an extract data set from the CDS and creating a report from it with the report utility.
- EDGBKUP is used to back up and recover the control data set and back up the journal.
- EDGAUD and EDGRPTD can help you get information about your removable media library and storage locations. You can also get security-related information about volumes and data sets defined to DFSMSrmm, and audit trail information about volumes, shelf assignments, and user activity.
- EDGUTIL is the utility used to create, update, and verify the CDS.
- EDGINERS is the DFSMSrmm utility that helps you erase and initialize tape volumes either automatically or manually. You can use EDGINERS to replace the DFSMSdfp utility IEHINITT.

You need to create a sample schedule of how frequently to run DFSMSrmm utilities. For more detailed information about the DFSMSrmm utilities, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.1.18 Running the installation verification program

The quickest and simplest way of getting DFSMSrmm started and verifying that the key pieces are installed successfully is to run the installation verification procedure (IVP). You can run the IVP at any time, for example, after installing maintenance on your system.

See the *z/OS Program Directory,* GI11-9848, for the IVP procedures. The information includes considerations for running in an environment where another tape management system is in use.

3.2 Tailoring the DFSMSrmm ISPF dialog

We describe how to make the DFSMSrmm ISPF panels available to the user, and how to modify only the panel layout for users and librarians.

To make the DFSMSrmm ISPF dialog available in the ISPF Master Application Menu, use one of the following techniques:

- Concatenate the DFSMSrmm target ISPF libraries with your existing ISPF libraries and use one of these methods:
  - Add DFSMSrmm to the ISPF dialog as explained in 3.2.1, “Adding DFSMSrmm to ISPF panels” on page 135.
  - Use the default method that is supplied by DFSMS. Select option R from the Interactive Storage Management Facility (ISMF) primary option menu.
  - Use the RMMISPF EXEC to invoke the dialog.
- Use the ISPF LIBDEF facility to make the DFSMSrmm target ISPF libraries available to your users, and use the EDGRMLIB EXEC to enter the dialog.
- When using the LIBDEF facility, if you are using different target library names as listed in the *z/OS V2R1 Program Directory*, GI11-9848, you must modify the EDGRMLIB EXEC or produce your own similar REXX exec or CLIST. For more information, see this PDF:

  http://www-03.ibm.com/systems/z/os/zos/library/bkserv/v2r1pdf/#EOZ

3.2.1 Adding DFSMSrmm to ISPF panels

You can add the DFSMSrmm selection in the ISPF Main Application Menu, so that users can choose it easily. To add the selection, follow these steps:

1. Add the DFSMSrmm selection to the body of the chosen ISPF selection panel. Add the text that is shown in Figure 3-78.

```
RM - RMM DFSMSrmm dialog
```

Figure 3-78 DFSMSrmm selection in an ISPF selection panel

2. Add one of the following statements to the ZSEL processing list in the PROC section of the chosen ISPF panel:
   - If you are not using LIBDEF, add the selection that is shown in Figure 3-79.

```
R,'CMD(%RMMISPF) NEWAPPL(EDG)'
```

Figure 3-79 Start RMM dialog using a new application

   - If you are using LIBDEF, add the selection that is shown in Figure 3-80.

```
R,'CMD(%EDGRMLIB)'
```

Figure 3-80 Start RMM dialog without a new application

Figure 3-81 on page 136 shows an example definition using LIBDEF.

```
...
$AO @AOC     @- Automated Operations Control/MVS Dialogs      +
$RM @RMM     @- DFSMSrmm dialog                               +
$CN @CONS    @- Console Display and Search Facility           +
...
)PROC
&ZSEL =TRANS(
             ..
             AO,'CMD(%$AOC140)'
             RM,'CMD(%EDGRMLIB)'
             ..
....
```

Figure 3-81 Adding DFSMSrmm to an ISPF selection panel

3.2.2 Modifying the DFSMSrmm panel

When you enter the DFSMSrmm ISPF dialog, the first panel that you usually see is the Primary Option Menu panel. You can change the panel navigation for the different types of users to go directly to a lower-level panel. For example, when they enter DFSMSrmm, tape librarians always see the Librarian Menu and users see the DFSMSrmm User Menu. You can modify the selection:

- You can modify the selection so that only the USER option of the DFSMSrmm dialog is available to the majority of users. Choose one of the following ways to change:
  - When using LIBDEF, see Figure 3-82 for an example of how to add the selection to the PROC section.

    ```
    R,'CMD(%RMMISPF USER) NEWAPPL(EDG)'
    ```

    Figure 3-82 ISPF PROC section

  - When using LIBDEF, modify the supplied EDGRMLIB EXEC to include the USER parameter on the RMMISPF EXEC call. For example, enter the REXX statement to replace the one that is supplied in EDGRMLIB as shown in Figure 3-83.

    ```
    address "ISPEXEC""SELECT CMD(%RMMISPF USER) NEWAPPL(EDG)PASSLIB"
    ```

    Figure 3-83 ISPEXEC call using a new application

If you are not using LIBDEF, Table 3-7 shows the names of the default libraries that you concatenate to the DD statements in the TSO logon procedure or a user-supplied start-up CLIST. If you are using LIBDEF, the EDGRMLIB EXEC allocates these default libraries to the user. You need to modify the EDGRMLIB exec to contain the new library names, after you change the names of the target libraries on your system.

Table 3-7 DFSMSrmm libraries and their default DD names

DFSMSrmm data set name	DD statement	Content
SYS1.SEDGEXE1	SYSPROC (or SYSEXEC)	REXX execs
SYS1.SEDGMENU	ISPMLIB	English messages
SYS1.SEDGPENU	ISPPLIB	English panel
SYS1.DGTSLIB	ISPSLIB	Skeletons
SYS1.DGTTLIB	ISPTLIB	Tables

DFSMSrmm data set name	DD statement	Content
SYS1.HELP	SYSHELP	TSO HELP

Note: The DFSMSrmm online help or the RMM TSO command is moved from SYS1.SEDGHLP1 to SYS1.HELP in z/OS V1.3.

- You can modify the selection so that the LIBRARIAN option of the DFSMSrmm dialog is available to the librarians. Choose one of the following methods to change it:
  - When using LIBDEF, add the following selection to the PROC section as shown in Figure 3-84.

```
R,'CMD(%RMMISPF LIBRARIAN) NEWAPPL(EDG)'
```

Figure 3-84 ISPF PROC section to call the library ISPF menu directly

  - When using LIBDEF, modify the supplied EDGRMLIB EXEC to include the USER parameter on the RMMISPF EXEC call. For example, enter the following REXX statement to replace the one supplied in EDGRMLIB as shown in Figure 3-85.

```
address "ISPEXEC""SELECT CMD(%RMMISPF LIBRARIAN) NEWAPPL(EDG)PASSLIB
```

Figure 3-85 ISPEXEC call for the librarian using a new application

3.2.3 Other changes

There are other changes that you might want to implement in the DFSMSrmm ISPF dialog:

- Add local dialog

  DFSMSrmm has a dummy panel named EDGP@LCL, which provides easy access to local dialog extensions. You can use these extensions without having to modify DFSMSrmm.

- Modify add product volume defaults

  You can modify the dialog EXEC EDGRPADV to change the values to ones that suit your installation.

Suggestion: Install your changes with a System Modification Program Extended (SMP/E) for z/OS USERMOD after editing a copy of the EDGRPADV EXEC.

- Modify DFSMSrmm messages

  DFSMSrmm provides messages for report titles and user notification. You can customize them by using these methods:

  - Updating the text of a message in the DFSMSrmm message table EDGMTAB
  - Applying changes to EDGMTAB by creating an SMP/E-installable USERMOD

For more information, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

3.3 Making TSO HELP information available to users

DFSMSrmm provides online help for the RMM TSO command. The help information is, by default, installed into the SYS1.HELP data set, although this can be changed by you during the SMP/E installation process.

You can update the TSO logon procedure or use the TSO ALLOCATE command. Example 3-14 shows how to update the TSO logon procedure.

Example 3-14 Update the TSO logon procedure

```
//IKJACCNT PROC
//IKJACCT  EXEC PGM=IKJEFT01, PARM='ISPPDF',DYNAMNBR=150,TIME=1440
//SYSPROC  DD   DSN=SYS1.OS390.CLIST,DISP=SHR
//SYSHELP  DD   DSN=SYS1.HELP,DISP=SHR
//*        DD   DSN=SYS1.SEDGHLP1,DISP=SHR
//ADMPC    DD   DSN=GDDM.SADMPCF,DISP=SHR
//ADMCGM   DD   DSN=GDDM.SADMSAM,DISP=SHR
//ADMGGMAP DD   DSN=GDDM.SADMMAP,DISP=SHR
//ADMPROJ  DD   DSN=GDDM.SADMMAP,DISP=SHR
//ADMIMG   DD   DSN=GDDM.SADMMAP,DISP=SHR
//ADMGIMP  DD   DSN=GDDM.SADMMAP,DISP=SHR
//ADMGDF   DD   DSN=GDDM.SADMMAP,DISP=SHR
//ADMSYMBL DD   DSN=GDDM.SADMSYM,DISP=SHR
//SYSPRINT DD   TERM=TS,SYSOUT=*
//SDFSDUMP DD   SYSOUT=*
//SYSTERM  DD   TERM=TS,SYSOUT=*
//SYSIN    DD   TERM=TS
//*
```

Note: The DFSMSrmm online help or the RMM TSO command is moved from SYS1.SEDGHLP1 to SYS1.HELP in z/OS V1.3.

3.4 Verifying your DFSMSrmm implementation

To test DFSMSrmm, you can use a small, test CDS or a converted CDS. We suggest that you use simple DFSMSrmm function tests for your first testing. Later, after you have analyzed the data and customized the exits and parameters, you can test those functions that are used in production.

Verifying DFSMSrmm functions by submitting batch jobs is simple. This section shows sample batch jobs that we used to verify DFSMSrmm functions in our system. Run the sample jobs in the sequence that is shown to test basic DFSMSrmm functions in your system.

3.4.1 Displaying the parmlib options and control information

Use the RMM LISTCONTROL subcommand to display information from the DFSMSrmm CDS control record and options that are defined in the EDGRMMxx parmlib member.

For example, display your installation options and rules, restricting the information that is displayed to the control record information and system options only (see Figure 3-86 on page 139). Specify the ALL operand to display all options and rules that are defined to your installation.

```
//LISTCNTL JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//* ************************************************************** *
//*          STEP 1  DISPLAY INFORMATION                           *
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM LISTCONTROL ALL
/*
```

Figure 3-86 Sample JCL to display parmlib options and control information

Example 3-15 shows the output from the RMM LISTCONTROL command.

Example 3-15 Sample RMM LISTCONTROL subcommand output

```
READY
 RMM LC ALL
Control record:
Type = MASTER    Create date = 2009/309    Create time = 16:38:11
                 Update date = 2012/287    Update time = 15:51:46
Journal: Utilization =   4% (75% threshold)     STATUS: = ENABLED
CDS:     Utilization =  19%
Exit status:                            Options:
  EDG_EXIT100 = ENABLED                   Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                      Extended Bin            = DISABLED
  EDG_EXIT300 = ENABLED                   Common Time             = DISABLED
                                          CDSID ENQ name          = ENABLED
Last backup:                            Last expiration processing:
  Date = 2012/130    Time = 22:07:52      Date = 2012/130    Time = 22:07:49
Last journal backup:                    Last store update:
  Date = 2012/130    Time = 22:07:52      Date =             Time =
Last report extract:                    Last VRS processing:
  Date = 2012/130    Time = 22:07:49      Date = 2012/130    Time = 22:07:49
Last scratch procedure:                 Last Catalog synchronize:
  Date =             Time =               Date =             Time =
Rack numbers       = 0                  Empty racks        = 0
LOCAL store bins   = 0                  Empty LOCAL bins   = 0
DISTANT store bins = 0                  Empty DISTANT bins = 0
REMOTE store bins  = 0                  Empty REMOTE bins  = 0
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =

 System options:
PARMLIB Suffix  = 02
Operating mode  = P    Retention period: Default = 0       Maximum = NOLIMIT
                                         Catalog = 6     hours
                       Retention method: Method  = VRSEL
Control data set name      = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold          = 75%               Journal transaction = NO
Catalog SYSID              = Notset
```

```
Scratch procedure name     = EDGXPROC
Backup procedure name      = EDGCDSBK
IPL date check  = N        Date format    = J       RACF support   = N
SMF audit       = 0        SMF security   = 42      CDS id         = SC70
MAXHOLD value   = 100      Lines per page = 54      System ID      = SC70
BLP             = RMM      TVEXT purge    = RELEASE Notify         = N
                                 days     = 0
Uncatalog       = Y        VRS job name   = 2       Message case   = M
MASTER overwrite= USER     Accounting     = J       VRS selection  = NEW
VRS change      = INFO     GDG duplicate  = BUMP    GDG cycle by = GENERATION
VRSMIN action   = INFO     VRSMIN count   = 1
VRSDROP action  = INFO     VRSDROP count  = 0              percent = 10
VRSRETAIN action= INFO     VRSRETAIN count= 0              percent = 80
EXPDTDROP action= INFO     EXPDTDROP count= 0              percent = 10
Disp DD name    = DISPDD   Disp msg ID    = EDG4054I
Retain by       = SET      Move by        = SET     CMDAUTH Owner  = NO
PREACS          = NO       SMSACS         = NO      CMDAUTH Dsn    = YES
Reuse bin       = CONFIRMMOVE                       Media name     = 3480
                                                    Local tasks    = 10
PDA: ON
 Block count    = 255      Block size     = 27      Log            = ON
SMSTAPE:
  Update scratch = YES      Update command = YES     Update exits   = YES
  Purge          = ASIS
 Client/Server:
  Subsystem type = STANDARD Port           = 0
  Server                    Server tasks   = 0
   host name  =
   IP address =
```

Security classes:

```
Number  Name      SMF  MSG  Erase  Description
------  --------  ---  ---  -----  --------------------------------
10      UCL       Y    N    N      UNCLASSIFIED
75      IUO       Y    N    N      IBM INTERNAL USE ONLY
100     IC        Y    N    Y      IBM CONFIDENTIAL
125     ICR       Y    N    Y      IBM CONFIDENTIAL RESTRICTED
150     RIC       Y    Y    Y      REGISTERED IBM CONFIDENTIAL
```

Volume Pools:

```
Pool   System   RA Ty Expdt Pool     Media    Description
                CF pe check name     name
------ -------- -- -- ----- -------- -------- --------------------------------
                     Action Scratch Overwrite
                     ------ ------- ---------
NS*             N  R  Y              3590     NORBERTS SCRATCH POOL
                           MANUAL USER
THS*            N  S  N              3590     3592 POOL
                            AUTO   USER
VT*             N  S  N     $$DFLT   3592     VTFM 3592 POOL
                            AUTO   LAST
*               N  S  O              3480     DEFAULT VOLUME POOL
                            AUTO   USER
```

Mount/Fetch messages:

```
Message ID    ID  Volume Rack
------------  --- ------ ----
IEF233A       1   15     999
```

```
NO INSTALLATION DEFINED REJECT PREFIXES. PARTITION AND OPENRULE COMMANDS ARE IN
CONT:- USE

Location definitions:
Location Def Mgtype  Ltype  Priority AM Medianames
-------- --- ------  ------ -------- -- ------------------------------------
          N          AUTO     4800   Y
          N          MANUAL   4900   Y
DISTANT   N          STORE     200   Y
LOCAL     N          STORE     300   Y
REMOTE    N          STORE     100   Y
SHELF     N                   5000   Y
VTFM001   Y  NOBINS  HSTORE   2000   N  *

Media Information :
                                                        Replace Policy
Name     Media type   Recording format Capacity (MB) Perm  Temp  WMC   Age
-------- ------------ ---------------- ------------- ----- ----- ----- -----
VTFM       5 VT3590G2     5 VT3590G2          1000       0     0     0     0
VTFM       5 VT3590G2     0 *                 1000       0     0     0     0
VTFM       0 *            0 *                    0       0     0     0     0


Partition Entries:
                        SMT       NOSMT
Volume or Range Type  Action Action Location
--------------- ----- ------ ------ --------
*               RMM   ACCEPT ACCEPT SHELF
*               NORMM IGNORE IGNORE

Openrule Entries:
                           Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   ACCEPT             ACCEPT
*               NORMM REJECT             REJECT

Action   Status
-------- ---------
REPLACE  Pending
SCRATCH  Pending
INIT     Unknown
ERASE    Unknown
RETURN   Unknown
NOTIFY   Unknown

Source   Target   Move Type Status
-------- -------- --------- ---------
SHELF    ATVIGA   NOTRTS    Pending
END
```

3.4.2 Adding owner information

Use the ADDOWNER subcommand to define an owner to DFSMSrmm. An *owner* can be an individual or group that is defined by a RACF group name, or any value you choose. The owner value contains 1 - 8 alphanumeric characters.

DFSMSrmm automatically creates an owner record if a user who is not defined to DFSMSrmm requests a job that writes to a volume managed by DFSMSrmm. DFSMSrmm uses the user ID that requests the job as an DFSMSrmm owner ID. In all other cases, the owner must be defined manually, and must be defined before you can use an RMM TSO subcommand that includes owner information.

Use the LISTOWNER subcommand to display information about a single owner that is defined to DFSMSrmm. For example, add details of a new owner and request information recorded by DFSMSrmm about the owner whose owner ID is SCHLUM (see Figure 3-87).

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
RMM ADDOWNER SCHLUM                                          -
               FNAME('NORBERT')                              -
               SNAME('SCHLUMBERGER')                         -
               DEPARTMENT('International Projects (A047)')     -
               ADD1('IBM Deutschland GmbH')                  -
               ADD2('IBM Allee 1')                           -
               ADD3('D-71139 Ehningen')                      -
               INTEL('919-3579')                             -
               EXTEL('++49-(0)-171-2222663')                 -
               USER(SCHLUM)                                  -
               NODE(IBMDE)                                   -
               EMAIL(SCHLUM§DE.IBM.COM

 RMM LISTOWNER SCHLUM
/*
```

Figure 3-87 Sample JCL to add owner information

Example 3-16 shows the output of the TSO RMM LISTOWNER subcommand.

Example 3-16 Sample RMM LISTOWNER subcommand output

```
Last name   = SCHLUMBERGER            First names = NORBERT
Department  = International Projects (A047)
Address     = IBM Deutschland GmbH
              IBM Allee 1
              D-71139 EHNINGEN
Telephone:
Internal    = 919-3579                External    = 49-(0)-171-2222663
Electronic mail:
Userid      = SCHLUM                  Node        = IBMDE
Email       = SCHLUM@DE.IBM.COM

Volumes     = 0

Last Change information:
Date             = 11/10/2012  Time = 07:00:20  System  = EGZB
User change date = 16/09/2003  Time = 17:34:49  User ID = *HKP
```

3.4.3 Adding racks to DFSMSrmm

The ADDRACK subcommand defines shelf locations in the removable media library and storage locations. DFSMSrmm defines shelf spaces in the removable media library as rack numbers and bin numbers in storage locations. Use the ADDRACK subcommand to define empty, or available, rack and bin numbers to DFSMSrmm. You must supply an initial rack number.

Note: There is no longer a need to add racks unless you want to assign racks to simplify management for volumes with volume numbers that are not in sequence.

The SEARCHRACK subcommand creates a list of shelf locations that are defined in the removable media library. Shelf locations in the removable media library are called *rack numbers*.

For example, add 20 racks to the removable media library, starting with rack number `SC0000`. Next, create a list of the created rack numbers (see Figure 3-88).

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM ADDRACK NS0000 -
             LOCATION(SHELF) -
             MEDIANAME(3592) -
             COUNT(15)

RMM SEARCHRACK RACK(NS*) LIMIT(*)
/*
```

Figure 3-88 Sample JCL to add RACK information

Example 3-17 shows the output of the TSO RMM SEARCHRACK subcommand.

Example 3-17 Sample RMM SEARCHRACK subcommand output

```
Rack/Bin          Media              Current  Moving-in  Moving-out  Old
number   Location name      Status   volume   volume     volume      volume
-------- -------- --------  -------  -------  ---------  ----------  ------
NS0000   3592     NS0000    EMPTY    SHELF
NS0001   3592     NS0001    EMPTY    SHELF
NS0002   3592     NS0002    EMPTY    SHELF
NS0003   3592     NS0003    EMPTY    SHELF
NS0004   3592     NS0004    EMPTY    SHELF
NS0005   3592     NS0005    EMPTY    SHELF
NS0006   3592     NS0006    EMPTY    SHELF
NS0007   3592     NS0007    EMPTY    SHELF
NS0008   3592     NS0008    EMPTY    SHELF
NS0009   3592     NS0009    EMPTY    SHELF
NS0010   3592     NS0010    EMPTY    SHELF
NS0011   3592     NS0011    EMPTY    SHELF
NS0012   3592     NS0012    EMPTY    SHELF
NS0013   3592     NS0013    EMPTY    SHELF
NS0014   3592     NS0014    EMPTY    SHELF
15         ENTRIES LISTED
```

3.4.4 Adding volumes to DFSMSrmm

The ADDVOLUME subcommand adds one or more volumes to DFSMSrmm. You have to define all volumes that you use in the DFSMSrmm CDS. DFSMSrmm monitors all tape volume mounts and automatically updates information about your predefined tape volumes when they are used.

The SEARCHVOLUME subcommand creates a list of volumes that matches the search criteria you specify. You can display lists of volumes according to ownership, assigned date, status, movement, action, pool, or media name.

For example, add 10 new scratch volumes in STEP01 to the removable media library using a rack number that is the same as the VOLSER. The volumes are not initialized. Then, add 10 new scratch volumes to the removable media manager into the racks. However, in this case, the volumes must be labeled before they can be used, as indicated by INITIALIZE(Y). Finally, create a list of all previously added volumes (see Figure 3-89).

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM ADDVOLUME NS0000                      -
               STATUS(SCRATCH)             -
               INITIALIZE(N)               -
               LOCATION(SHELF)             -
               MEDIANAME(3592)             -
               MEDIATYPE(*)                -
               COUNT(10)

 RMM ADDVOLUME NS0010                      -
               STATUS(SCRATCH)             -
               INITIALIZE(Y)               -
               LOCATION(SHELF)             -
               MEDIANAME(3592)             -
               MEDIATYPE(*)                -
               POOL(SC*)                   -
               COUNT(5)

 RMM SEARCHVOLUME VOLUME(NS*) OWNER(*) LIMIT(*)
/*
```

Figure 3-89 Sample JCL to add volume information

Note: In the second example, we will add the volumes using a different pool prefix. In this case, you have to define rack numbers first.

Example 3-18 shows the output of the TSO RMM SEARCHVOLUME subcommand.

Example 3-18 Sample RMM SEARCHVOLUME subcommand output

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
NS0000          NS0000 2010/289              SHELF    0     S
NS0001          NS0001 2010/289              SHELF    0     S
NS0002          NS0002 2010/289              SHELF    0     S
NS0003          NS0003 2010/289              SHELF    0     S
NS0004          NS0004 2010/289              SHELF    0     S
```

```
NS0005          NS0005 2010/289              SHELF    0     S
NS0006          NS0006 2010/289              SHELF    0     S
NS0007          NS0007 2010/289              SHELF    0     S
NS0008          NS0008 2010/289              SHELF    0     S
NS0009          NS0009 2010/289              SHELF    0     S
NS0010          SC0000 2010/289              SHELF    0     I  I
NS0011          SC0001 2010/289              SHELF    0     I  I
NS0012          SC0002 2010/289              SHELF    0     I  I
NS0013          SC0003 2010/289              SHELF    0     I  I
NS0014          SC0004 2010/289              SHELF    0     I  I
```

3.4.5 Adding bins to DFSMSrmm

The ADDBIN subcommand defines shelf locations in the storage locations. DFSMSrmm defines shelf spaces in storage locations as bin numbers. Use the ADDBIN subcommand to define empty bin numbers to DFSMSrmm.

The SEARCHBIN subcommand creates a list of shelf locations that are defined in the storage locations. Shelf locations in the storage locations are called *bin numbers*.

For example, add five empty shelf locations to the built-in storage location SANJOSE, starting with bin number 000001 (see Figure 3-90). The example assumes that you have defined storage location SANJOSE with the LOCDEF command in parmlib member EDGRMMxx. The second command creates a list of the new bin numbers.

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM ADDBIN 000001            -
            LOCATION(SANJOSE) -
            MEDIANAME(3592)   -
            COUNT(5)

 RMM SEARCHBIN BIN(*) LIMIT(*) LOCATION(SANJOSE)
/*
```

Figure 3-90 Sample JCL to add bins

Example 3-19 shows the results of the TSO RMM SEARCHBIN subcommand.

Example 3-19 Sample RMM SEARCHBIN subcommand output

```
Rack/Bin           Media              Current  Moving-in  Moving-out Old
number   Location  name      Status   volume   volume     volume     volume
-------- --------  --------  -------  -------  ---------  ---------- ------
000001   SANJOSE   3590      EMPTY
000002   SANJOSE   3590      EMPTY
000003   SANJOSE   3590      EMPTY
000004   SANJOSE   3590      EMPTY
000005   SANJOSE   3590      EMPTY
SEARCH COMPLETE - MORE ENTRIES MAY EXIST
5          ENTRIES LISTED
```

3.4.6 Adding a DSNAME VRS to DFSMSrmm

The RMM ADDVRS command adds a new VRS to DFSMSrmm. A VRS is used to define retention and movement policies for data sets and volumes.

The LISTVRS subcommand displays details about a single VRS. Specify a data set name when requesting information about a data set VRS.

For example, add a data set VRS to retain all data sets matching the DSNAME mask SCHLUM.RMMTEST.MOVE.** in location REMOTE until the data set is no longer cataloged (see Figure 3-91). The second command requests information about the data set VRS defined for SCHLUM.RMMTEST.MOVE.** data sets.

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM ADDVRS DSNAME('SCHLUM.RMMTEST.MOVE.**')         -
            CYCLES COUNT(99999) STORENUMBER(1)       -
            DESCRIPTION('RETAIN AND MOVE DATA SETS') -
            LOCATION(REMOTE)                         -
            OWNER(SCHLUM)                            -
            WHILECATALOG

 RMM LISTVRS DSNAME('SCHLUM.RMMTEST.MOVE.**')
/*
```

Figure 3-91 Sample JCL to add VRS

Example 3-20 shows the result of the TSO RMM LISTVRS subcommand.

Example 3-20 Sample RMM LISTVRS subcommand output

```
 RMM LISTVRS DSNAME('SCHLUM.RMMTEST.MOVE.**')
Data set mask = SCHLUM.RMMTEST.MOVE.**                      Type = DSNAME
Job name mask =                              Retain until expired = NO
Count        = 99999 CYCLES                 Retain while cataloged = YES
Delay        = 0     Days in the HOME location
Store number = 1     CYCLES  in the REMOTE   location
Priority     = 0                                  Release Options:
                                                Expiry date ignore = NO
                                                 Scratch immediate = NO
Next VRS in chain    =          using     VRS

VRS Owner    = SCHLUM
Description  = RETAIN AND MOVE DATA SETS
Last Reference: Date =            Time =
Vital Record Specification to be deleted on 1999/365

Last Change information:
Date             = 2010/289    Time = 20:45:17  System  = SC70
User change date = 2010/289    Time = 20:45:17  User ID = MHLRES7
```

3.4.7 Creating tape data sets

Use the IEBGENER utility to create a multifile, multivolume chain to understand the differences between DFSMSrmm and your tape management system. Figure 3-92 on page 147 shows sample JCL for creating different data sets on tape.

```
//*        ************************************************************
//*        * 1. STEP   ALLOC 1. FILE  DSN=SCHLUM.RMMTEST.FILE1.VOL1   *
//*        *           VOLUME 1 / FILE 1 / RETPD 5480 (APPROX. 15 Y.) *
//*        ************************************************************
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DSN=DFRMM.TEST.DATA.SMALL,DISP=SHR
//SYSUT2   DD   DSN=SCHLUM.RMMTEST.FILE1.VOL1,
//         DISP=(,CATLG,DELETE),
//         UNIT=VT3590,VOL=(,RETAIN),
//         DCB=*.SYSUT1,LABEL=RETPD=5480
//SYSIN    DD   DUMMY
...
//*        ************************************************************
//*        * 5. STEP   ALLOC 5. FILE  DSN=SCHLUM.RMMTEST.FILE5.VOL3   *
//*        *           VOLUME 3 / FILE 2 / RETPD 10                   *
//*        ************************************************************
//STEP05   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DSN=DFRMM.TEST.DATA.SMALL,DISP=SHR
//SYSUT2   DD   DSN=SCHLUM.RMMTEST.FILE5.VOL3,DISP=(,CATLG,DELETE),
//         DISP=(,CATLG,DELETE),
//         DCB=*.SYSUT1,LABEL=(5,SL,,RETPD=10),
//         VOL=(,RETAIN,,REF=*.STEP04.SYSUT2)
//SYSIN    DD   DUMMY
```

Figure 3-92 Sample JCL to create data sets on tape

3.4.8 Allocating inventory management data sets

Before you can use the EDGHSKP utility to run inventory management activities, you must define several data sets, but not all data sets are required for all functions. Some data sets used during inventory management must be pre-allocated and cataloged because these data sets are used by both the EDGHSKP utility and the DFSMSrmm subsystem. For additional information and sample jobs, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Figure 3-93 shows the sample JCL that we used to allocate these data sets.

```
//STEP01   EXEC PGM=IEFBR14
//MESSAGE  DD   DSN=RMM.PROD.MSGS,DISP=(NEW,CATLG),
//         UNIT=SYSDA,AVGREC=U,SPACE=(4096,(10,10))
//REPORT   DD   DSN=RMM.HSKP.REPORT.FILE.NAME,DISP=(NEW,CATLG),
//         LRECL=137,RECFM=VBA,
//         UNIT=SYSDA,AVGREC=U,SPACE=(4096,(50,30))
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY.FILE.NAME,DISP=(NEW,CATLG),
//         UNIT=SYSDA,AVGREC=U,SPACE=(4096,(50,30))
//REPTEXT  DD   DSN=RMM.HSKP.REPORT.EXTRACT.FILE.NAME,DISP=(NEW,CATLG),
//         UNIT=SYSDA,AVGREC=U,SPACE=(4096,(1000,500))
//XREPTEXT DD   DSN=RMM.HSKP.REPORT.XEXTRACT.FILE.NAME,DISP=(NEW,CATLG),
//         UNIT=SYSDA,AVGREC=U,SPACE=(4096,(1000,500))
```

Figure 3-93 Sample JCL to pre-allocate EDGHSKP data sets

The REPTEXT and XREPTEXT DD statements allocate data sets that contain the extract copy of the DFSMSrmm control data set. The extract copy is called the *extract data set*. You must specify either the REPTEXT DD statement or the XREPTEXT DD statement when you use the EDGHSKP RPTEXT parameter. You do not need both data sets created. A single REPTEXT data set can be used for either normal extract or extended extract.

You can also use the sample JCL in SAMPLIB member EDGJHKPA to allocate the ACTIVITY, MESSAGE, REPORT, and RPTEXT files that are required to run inventory management functions.

3.4.9 Running inventory management

After you create racks, volumes, VRSs, and data sets, you must run inventory management:

1. Run vital record processing (VRSEL) before expiration and storage location management processing to identify which volumes to retain and where volumes need to be moved according to their VRSs.
2. Run storage location management (DSTORE) to set a destination location, and optionally assign shelf locations in storage locations to volumes that are being sent out of the removable media library.
3. Run expiration processing (EXPROC) to identify the volumes that are not required for vital records that are ready to expire. During expiration processing, release actions for volumes are noted.
4. Run extract data set processing (RPTEXT) to a sequential data set that contains information from DFSMSrmm and use it as input to EDGRPTD, the DFSMSrmm reporting utility.
5. Back up the CDS and optionally the journal (BACKUP), and reset the journal data set when the CDS and journal are backed up using the EDGHSKP utility.

Figure 3-94 on page 149 shows sample JCL to perform inventory management and to copy the message data set to your job log.

```
//STEP01   EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RMM SEARCHVOLUME VOLUME(*) ACTION(SCRATCH) LIMIT(*) OWNER(*)
/*
//STEP02   EXEC PGM=EDGHSKP,
//         PARM='VRSEL,DSTORE,EXPROC,RPTEXT,BACKUP,DATEFORM(J)'
//SYSPRINT DD   SYSOUT=*
//MESSAGE  DD   DSN=RMM.PROD.MSGS,DISP=SHR
//REPORT   DD   DSN=RMM.HSKP.VRSEL,DISP=SHR
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY,DISP=SHR
//REPTEXT  DD   DSN=RMM.HSKP.EXTRACT,DISP=SHR
//BACKUP   DD   DSN=RMM.HSKP.BACKUP,DISP=(,CATLG,DELETE),
//         SPACE=(TRK,(150,10),RLSE),
//         DCB=(GDG,LRECL=9216,BLKSIZE=0,RECFM=VB)
//JRNLBKUP DD   DSN=RMM.HSKP.JRNLBKUP,DISP=(,CATLG,DELETE),
//         SPACE=(TRK,(10,10),RLSE),
//         DCB=(GDG,LRECL=9248,BLKSIZE=0,RECFM=VB)
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD   DUMMY
//SYSUT1   DD   DSN=RMM.PROD.MSGS,DISP=SHR
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 3-94 Sample JCL to run inventory management

3.4.10 Confirming movements

Confirm your outstanding volume moves by using the RMM CHANGEVOLUME command. In DFSMSrmm, confirm each volume move, except for volumes that are moved to an IBM tape library, to confirm that you moved the volume.

You can confirm a move in two ways:

- For a single volume, use the RMM TSO CHANGEVOLUME subcommand as shown in Figure 3-95 with a VOLSER and the CONFIRMMOVE operand.

  No further actions are required.

```
RMM CHANGEVOLUME vvvvvv CMOVE
```

Figure 3-95 RMM confirm move for a single volume

- Figure 3-96 shows a global confirmation of volume movements. Specify an asterisk (*) for the VOLSER and ALL for the from and to location.

  In this case, you must rerun the storage location management processing to update the DFSMSrmm CDS.

```
RMM CHANGEVOLUME * CMOVE(ALL,ALL)
or
RMM CHANGEVOLUME * CMOVE(ALL,SANJOSE)
RMM CHANGEVOLUME * CMOVE(SANJOSE,ALL)
```

Figure 3-96 RMM global confirm move

3.4.11 Creating reports

You can create several types of reports with the DFSMSrmm report utilities, EDGRPTD and EDGAUD, and also by using the DFSMSrmm report generator. DFSMSrmm provides standard reports and samples shipped in SAMPLIB. You can use the sample DFSMSrmm EDGJRPT JCL that invokes the EDGRRPTE exec to create the reports. The input to the reporting execs is the extended extract data set. Documentation about the reports is shipped in SYS1.SAMPLIB in the EDGDOC file.

Use the reports that are created to help you keep track of your DFSMSrmm environment, and compare these reports with those of your tape management system running in parallel. You can use the DFSORTS ICETOOL to write customized reports, or you can use the DFSMSrmm TSO subcommands to create lists of information that is defined to DFSMSrmm.

Creating reports with EDGRPTD

Use EDGRPTD to create inventory reports for auditing the physical contents of the installation media library and storage locations. You can also use EDGRPTD to create movement reports that list volumes to be moved from one location to another. Use these reports to make an inventory of your volumes and to identify volumes that need to be pulled and moved to other locations.

DFSMSrmm-supplied reports

DFSMSrmm provides REXX execs and JCL in SYS1.SAMPLIB member EDGJRPT that you can use as-is to create the following reports:

- Pull list for scratch tapes sorted by volume serial number
- Pull list for scratch tapes sorted by data set name
- List of all volumes in scratch status
- List of volumes that are ready to scratch
- Ready to scratch
- Inventory list by volume serial number
- Inventory list by data set name
- Inventory of data sets including number of KBs used
- Inventory of volume serial numbers by location
- Inventory of data set names by location
- Inventory of BIN numbers by location
- List of all data set names at loan location
- List of all volume serial numbers at loan location
- List of all multiple volume, multiple data sets
- Movement report including the first data set name on the volume
- Movement report sorted by storage location bin number
- Movement report sorted by volume serial number
- Inventory list sorted by volume serial number including volume count
- Inventory of duplicate volumes
- Inventory of stacked volumes by percent active
- Inventory of data sets by volume retention method

You can copy these reports and use them to create reports that are tailored for your installation.

Creating reports with EDGAUD

Use EDGAUD to create security and audit reports using either previously selected and sorted SMF records or raw SMF data. DFSMSrmm produces SMF records when you specify the SMFAUD or SMFSEC installation options in the parmlib member EDGRMMxx. DFSMSrmm uses the current subsystem start-up option for the SMF record types and default report option unless you override them with the EDGAUD EXEC parameters.

Creating reports with the report generator

The report generator guides you through ISPF panels to define your reports and then it produces JCL. The generated JCL includes all the commands that are needed for producing the report as you specified. To obtain the report, you must submit the JCL. You can use this JCL for subsequent reports.

For detailed information about how to create a report, see Chapter 14, “Report Generator” on page 573.

3.4.12 Running EDGINERS to initialize volumes automatically

With EDGINERS, volumes can be initialized and erased, based on the volume status and actions recorded in the CDS, or you can let your users or operator perform the processing. For more information about operator tasks, such as responding to initialization messages, tape mount messages, and using the LABEL procedure to request EDGINERS processing, see *DFSMSrmm Guide and Reference,* SC26-7404. For a description of the EDGLABEL procedure that is provided by DFSMSrmm, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

During automatic processing, the process requests each volume to be mounted, as with any batch job, because an operator reply is not required. When the volume is mounted, processing continues. If the operator cannot mount a volume, DFSMSrmm allows them to skip processing the current requested volume by replying to a WTOR.

As part of volume expiration processing, DFSMSrmm records when volumes need to be initialized or erased as they are released for return to scratch. If you use the EXEC parameters in your JCL to set up automatic processing, DFSMSrmm initializes or erases those volumes without librarian intervention. EDGINERS issues WTOR messages and MSGDISP requests to the operator and drive to get a volume mounted and dismounted.

You can run EDGINERS automatic processing to initialize the volumes you are adding to DFSMSrmm. When you add scratch volumes to DFSMSrmm, use the RMM TSO subcommand operand, INIT(Y), on the ADDVOLUME command to ensure that volumes are initialized before they are available as scratch. Specify the initialization parameters on the PARM statement in your EDGINERS JCL. DFSMSrmm selects all volumes that are marked as requiring initialization and performs automatic processing. Figure 3-97 shows sample JCL for automatic processing.

```
//STEP01   EXEC PGM=EDGINERS,PARM='MEDIANAME(3592),VERIFY'
//SYSPRINT DD   SYSOUT=*
//TAPE     DD   UNIT=(VT3590,,DEFER)
```

Figure 3-97 Sample JCL for EDGINERS automatic processing

This job initializes the volumes NS0010 - NS0014 added in Figure 3-89 on page 144. The VERIFY parameter requests that DFSMSrmm ask the operator to remount each volume that has been successfully erased or labeled. The volumes are requested in reverse order, and the volume labels are read to ensure that operator errors have not occurred (for example, a mismatch between the internal label and the external label).

3.4.13 Restoring the CDS

To restore the CDS, you first delete the existing CDS and then redefine it using Access Method Services (AMS) commands. The EDGBKUP utility is used to forward recover the CDS.

During the recovery, you must stop all DFSMSrmm activities with the MVS command that is shown in Figure 3-98.

```
MODIFY DFRMM,QUIESCE
```

Figure 3-98 Stop all tape activities

Important: In a multihost environment, issuing a DFSMSrmm quiesce affects only the host on which you issue the command. If you want all hosts to be quiesced, you must issue the command on each host that is sharing the control data set.

Using the backups of the CDS and journal that is taken by the EDGHSKP utility during inventory management, you can recover the DFSMSrmm CDS at any time. Figure 3-99 shows how you can use the latest backup of the CDS and the current journal file to restore and forward recover to the point of failure.

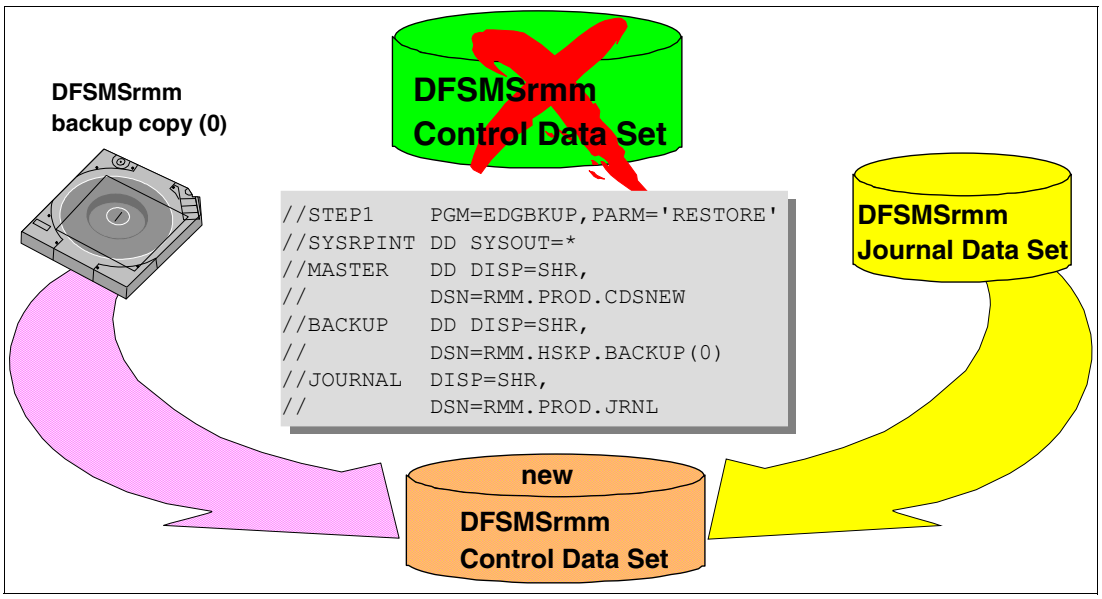


Figure 3-99 Recovering a CDS from the latest backup

If the latest backup of the CDS is not available, you can use an older backup and the backups of the journal to recover. Figure 3-100 shows how you can use the previous backup generation of the CDS, the latest journal backup, and the current journal file to restore and forward recover to the point of failure.

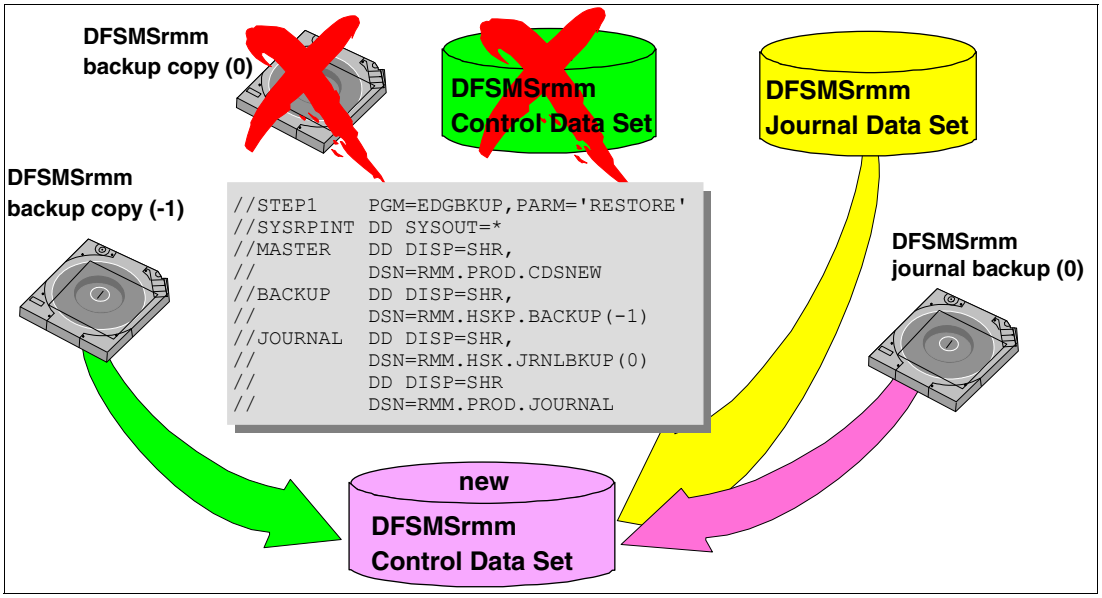


Figure 3-100 CDS recovering from backup copy (-1)

Using this process, you can recover from any CDS backup as long as you have the corresponding journal backups.

You can restore the DFSMSrmm CDS and forward recover the restored CDS with the journal. Use the JOURNAL DD statement to concatenate multiple journal data sets. If you forward recover using multiple journal data sets, ensure that the journal data sets are concatenated in the order in which changes were originally made.

When a backup is taken using DFSMSdss, by specifying BACKUP(DSS) for any backup that is being restored, a journal backup is also required for forward recovery. For the latest CDS backup, both the latest journal backup and the active journal must be used for forward recovery. In a remote recovery site, only the journal backup is required for successful forward recovery of the restored CDS.

In our example, we use a different data set name for the DFSMSrmm control data set than the original one. If you continue with the new restored CDS, you have to rename it or you must update the EDGRMMxx parmlib member to specify the new name in the DSNAME option.

Figure 3-101 on page 154 shows sample JCL to delete and define the CDS before the forward recovery.

```
//STEP01 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//MASTER DD DISP=SHR,UNIT=3390,VOL=SER=DFRMM4
//SYSIN DD *
  DELETE RMM.PROD.CDSNEW CL
  SET MAXCC = 0
/*
   DEFINE CLUSTER(NAME(RMM.PROD.CDSNEW)  -
          FILE(MASTER)                   -
          FREESPACE(15 0)                -
          KEYS(56 0)                     -
          REUSE                          -
          RECSZ(512 9216)                -
          SHR(3 3)                       -
          KILOBYTES(4500 1500)           -
          STORAGECLASS(gspace)           -
          DATACLASS(EFKSDS)              -
          VOLUMES(DFRMM4))               -
   DATA(NAME(RMM.PROD.CDSNEW.DATA)       -
          BUFFERSPACE(829440)            -
          CISZ(26624))                   -
   INDEX(NAME(RMM.PROD.CDSNEW.INDEX)     -
          CISZ(2048))
/*
```

Figure 3-101 Sample JCL to delete and define the CDS

Figure 3-102 shows sample JCL to forward recover the CDS for a backup with BACKUP(AMS). In this case, you have to specify a JOURNAL DD statement with the actual journal.

```
//STEP03   EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DSN=RMM.PROD.CDSNEW,DISP=SHR
//BACKUP   DD   DSN=RMM.HSKP.BACKUP(0),DISP=SHR
//JOURNAL  DD    DSN=RMM.PROD.JRNL,DISP=SHR
```

Figure 3-102 Sample JCL to forward recover the CDS (backup with AMS)

Figure 3-103 shows sample JCL to forward recover the CDS for a backup with BACKUP(DSS).

```
//STEP03   EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DSN=RMM.PROD.CDSNEW,DISP=SHR
//BACKUP   DD   DSN=RMM.HSKP.BACKUP(0),DISP=SHR
//JOURNAL  DD   DSN=RMM.HSKP.JRNLBKUP(0),DISP=SHR
//         DD   DSN=RMM.PROD.JRNL,DISP=SHR
```

Figure 3-103 Sample JCL to forward recover the CDS (backup with DSS)

Note: If you are using DSS, you must always specify the last backup copy of the journal and the journal.

You can check that restore and forward recovery were successful by displaying volume information. All volumes must have been initialized if the job shown in Figure 3-97 on page 151 ended without any errors.

When the CDS recovery is complete, restart DFSMSrmm using the MVS operator command that is shown in Figure 3-104.

```
MODIFY DFRMM,M=xx
```

Figure 3-104 Restart DFSMSrmm

Important: In a multihost environment, issuing a DFSMSrmm MODIFY command affects only the host on which you issue the command. If you want all hosts restarted, you must issue the command on each host that is sharing the control data set.

3.4.14 Verifying the CDS

Use the EDGUTIL utility to verify the contents of the CDS. You must perform the verification each time you recover your DFSMSrmm CDS. Figure 3-105 shows a sample JCL to verify the CDS.

```
//STEP01   EXEC PGM=EDGUTIL,PARM='VERIFY(ALL)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDSNEW
//SYSIN    DD   DUMMY
```

Figure 3-105 Sample JCL to verify the CDS contents

Important: The VERIFY(ALL) DFSMSrmm performs extra processing with the TCDB and library manager. That can be a very long running process. DFSMSrmm scans both the DFSMSrmm control data set and the TCDB sequentially to find DFSMSrmm volumes that are not in the TCDB, and TCDB volumes that are not in the DFSMSrmm control data set. DFSMSrmm also checks any volume that is found to be system-managed, either by definition to DFSMSrmm or retrieved from the TCDB, against the library manager database for an IBM automated tape library.

3.4.15 Testing implemented exits

If you use the exits that are presented in the following sections, you must test them.

EDGUX100

Use the DFSMSrmm EDGUX100 installation exit to perform these functions:

- Ignore foreign or duplicate volumes
- Use JCL special expiration dates to manage volumes by setting VRS management values to retain data sets
- Set any expiration date, including a zero value, for a data set

- Perform pooling management:
  - Manage scratch pools based on job name and data set name
  - Select a pool to be used for a non-specific tape volume request
  - Select a specific pool to be used for a non-specific tape volume request and request that the tape drive cartridge loader is disabled
- Pass the VRS management value and pooling decisions to pre-ACS processing that can be used in ACS routines to assign storage group and management class
- Obtain information to modify input from DFSMSrmm disposition processing
- Request the recording of only the first file on a multifile volume
- Change the location for a volume
- Support the use of the storage group name for pooling for volumes that reside in manual tape libraries

You can use the supplied sample EDGUX100 exit to support the special expiration dates 98000 and 99000 without modifications to the sample code. If you plan to support other special dates using the exit and have modified it, you also need to test them. The results and the testing of the additional dates are the same as for the 99000 date.

To test the use of the 99000 special date, create a new tape data set by specifying **EXPDT=99000**. Use the DFSMSrmm LISTDATASET command or the list data set dialog to display details of the data set that must include D99000 as the VRS management value.

To test the use of **EXPDT=98000** or **ACCODE=XCANORES** bypass tape management, first define a RACF FACILITY class resource (see Appendix A, “Security topics” on page 633) to allow the use of the ignore function to bypass DFSMSrmm recording. Whether you test to check that ignore is allowed or that ignore is not allowed, DFSMSrmm issues a message of explanation. To test the ignore function, DFSMSrmm must be running in either warning or protect mode.

EDGUX200

Use the DFSMSrmm EDGUX200 installation exit to perform these functions:

- Return a volume to scratch status in an external inventory
- Prevent a volume from returning to scratch status
- Request that DFSMSrmm ignores data set information that is recorded for a volume

The sample EDGUX200 provides no specific function. Test the exit only if you plan to use it and have modified the supplied source code. The exit is intended for communication with software that has an inventory of scratch tapes. To test the exit, you need a tape volume that is in pending release status and is to return to scratch. Run inventory management EXPROC processing and then check that the external inventory has been correctly updated.

System-managed tape support by using the OAM exits

Test the following exits only if you have an IBM system-managed tape library:

CBRUXENT	Cartridge entry exit
CBRUXEJC	Cartridge eject exit
CBRUXCUA	Cartridge change use attributes exit
CBRUXVNL	Volume not in library exit

DFSMSrmm provides sample exits that are ready to use, installed and active on your system, and do not require modification. However, you can customize the exits to meet your needs. CBRUXVNL is the exit you might want to consider customizing.

A simple set of tests can be used to test these exits:

1. Define a volume to DFSMSrmm for entry to the library using the command that is shown in Figure 3-106.

```
RMM ADDVOLUME VOLSER LOCATION(ATL) STATUS(MASTER)
```

Figure 3-106 RMM ADDVOLUME subcommand

   If you list or display the volume, it needs to be in MASTER status with a destination of ATL.

2. Enter the volume into the library:
   a. To test CBRUXENT, list or display the volume in DFSMSrmm. It now shows a location of ATL and no destination. If you display the volume using ISMF, the volume displays as private.
   b. To test CBRUXCUA, use ISMF to change the volume status to scratch. If DFSMSrmm is in protect mode, the change is rejected. If DFSMSrmm is in warning mode, the change is allowed, but a warning message is issued that DFSMSrmm only allows the change in warning mode. Messages are not issued in record or manual mode.
   c. To test CBRUXEJC, eject the volume from the library. You can use any method for the ejection. When the eject is completed and the volume displays in DFSMSrmm, the volume shows as being in transit.
   d. To test CBRUXVNL, ensure that the volume is in private or master status, and run a job to create or read a file on the tape.

      You see message EDG8124I that lists DFSMSrmm information or message EDG8123D on the console. After receiving EDG8124I, depending on the circumstances, either WTOR EDG8121D or EDG8122D appears on the console. Reply by entering `RETRY` after the volume is entered into the library, and the job runs successfully.

Checking that all OAM exits are enabled

Use the MVS command in Figure 3-107 to verify that all OAM exits are enabled.

```
D SMS,OAM
```

Figure 3-107 Displaying the OAM status

The result of the DISPLAY SMS command is shown in Example 3-21.

Example 3-21 Result of the display SMS,OAM command

```
F OAM,D,OAM,L=MHLRES7-Z
CBR1100I OAM status: 167
TAPE TOT  ONL  TOT  TOT  TOT  TOT    TOT    ONL    AVL    TOTAL
     LIB  LIB  AL   VL   VCL  ML     DRV    DRV    DRV   SCRTCH
       3    1    1    0    0    0     16      0      0        9
There are also 0 VTS distributed libraries defined.
CBRUXCUA processing ENABLED.
CBRUXEJC processing ENABLED.
CBRUXENT processing ENABLED.
CBRUXVNL processing ENABLED.
CBRUXSAE processing ENABLED.
CBRUXSAE processing ENABLED for STORE.
CBRUXSAE processing ENABLED for RETRIEVE.
CBRUXSAE processing ENABLED for QUERY.
CBRUXSAE processing ENABLED for CHANGE.
```

```
CBRUXSAE processing ENABLED for DELETE.
EDGTVEXT processing ENABLED.
Access Backup processing ACTIVE for UNREADABLE VOLUMES, using 1st
backup copy.
Access Backup processing ACTIVE for OFFLINE LIBRARIES, using 1st
backup copy.
Access Backup processing ACTIVE for NOT OPERATIONAL LIBRARIES, using
1st backup copy.
Access Backup processing ACTIVE for DB2 OBJECT TABLE ERRORS, using
1st backup copy.
Access Backup processing ACTIVE for LOST VOLUMES, using 1st backup
copy.
Access Backup processing ACTIVE for FILE SYSTEM ERRORS, using 1st
backup copy.
Diagnostic messages ACTIVE for OSREQFS. Limit=  10.
DB2 SSID:  D9DG
XCF GROUP NAME: OAMTEST
XCF MEMBER NAME: OAM70
CBROAM: 70
OAM1 Parms: TIME=LOC  MSG=EM  UPD=Y  QB=Y
            MOS=2000  OTIS=N  LOB=A  DP=A
```

3.4.16 Testing application use of tape

Understand the ways in which tape is used in your installation. Although we assume that everyone's use is the same, it is surprising how often we find new and odd uses of tape. To avoid discovering problems in production with DFSMSrmm, test the ways in which you use tapes today to ensure that DFSMSrmm supports them as expected. For example, you might use BLP (perhaps to scratch tapes) or non-labeled tapes, or you might have key business functions that depend on foreign tape processing.

Use tests that you have already in place for your production tape usage under DFSMSrmm to validate that they are supported and that you know which actions might be required to support them. For example, RACF profiles are required to use no label (NL) output to scratch and to process tapes that are to be ignored by DFSMSrmm.

NL scratch tapes

DFSMSrmm normally supports only standard label scratch tapes for non-specific NL tapes, and relabels them at time of use to NL and standard label at return to scratch. To use NL scratch tapes, you can use the system tape exit IFG019VM. DFSMSrmm now ships a sample IFG019VM exit as member EDG019VM in SAMPLIB. The sample exit checks for a supported request and issued a WTOR to the operator to obtain the volume serial number. Testing ensures that it does support the installation needs for non-specific NL tape.

Interface between DFSMShsm and DFSMSrmm

Test the built-in interface between DFSMShsm and DFSMSrmm. Before you start this kind of test, ensure that DFSMShsm has the correct authority for the STGADMIN resources in the RACF class FACILITY. If you have a release of DFSMSrmm earlier than OS/390 Version 1 Release 4.0, you must define the use of the ARCTVEXT installation exit in the DFSMShsm parmlib member. You can test the function of the EDGTVEXT interface, the use of the ARCTVEXT exit EDGTVEXT interface, or the use of the ARCTVEXT exit by deleting a volume from the DFSMShsm tape volume pool using the HSM DELVOL command. The TVEXTPURGE parmlib operand specifies how DFSMSrmm processes volumes to be purged by callers of EDGTVEXT or EDGDFHSM. It is the interface between DFSMShsm and DFSMSrmm.

There are three options that can be specified for TVEXTPURGE.

The following values apply:

NONE DFSMSrmm takes no action for volumes to be purged.

RELEASE DFSMSrmm releases volumes to be purged according to the release actions set for the volume. You must run expiration processing to return a volume to scratch status.

EXPIRE Use the EXPIRE option to set the volume expiration date to the current date for volumes to be purged.

The recommended option is TVEXTPURGE(RELEASE).

Suggested DFSMShsm system option parameters

Use the DFSMShsm system option parameters that are shown in Figure 3-108.

```
SETSYS EXITOFF(ARCTVEXT)        -
       SELECTVOLUME(SCRATCH)    -
       TAPEDELETION(SCRATCH)    -
       TAPESECURITY(EXPIRATION) -
       PARTIALTAPE(MARKFULL)
```

Figure 3-108 Suggested DFSMShsm system option parameters

Note: Specify EXITOFF(ARCTVEXT) if DFSMSrmm is your only tape management product because DFSMShsm always calls the DFSMSrmm programming interface EDGTVEXT.

3.5 Education

Consider which groups of users might be affected by the initial introduction of DFSMSrmm for testing. Your colleagues, the operators, and possibly the tape librarian need to learn about DFSMSrmm at some stage. At this stage, provide a minimum level of education so that any potential impact on your production system can be minimized by ensuring that affected users are aware of changes.

3.6 New operator procedures

When you start to run DFSMSrmm, system operators are most likely affected. DFSMSrmm can cause the following changes:

- Mount and fetch messages are updated.
- Scratch pool names might be different and are included in the MVS mount messages.
- Tape drive displays are updated with rack number and pool information.
- DFSMSrmm issues WTORs that require operator action and decisions.

In addition, you might want the operator to be involved in starting and stopping DFSMSrmm.

For details about messages and activities that affect the operator, see the *DFSMSrmm Guide and Reference*, SC26-7404.

4

DFSMSrmm retention methods

This chapter explains how to use the different retention methods Data Facility System Managed Storage removable media manager (DFSMSrmm) provides to retain your DFSMSrmm-managed tape volumes.

This chapter contains the following information:

- Overview
- Specifying a retention method:
  - Expiration date
  - Retention date
- Using the EDGRMMnn parmlib option RETENTIONMETHOD
- EDG_EXIT100 retention method support
- Subcommands for RETENTIONMETHOD parameters
- Syntax format of the RETENTIONMETHOD operand
- Using the SEARCHVOLUME subcommand

By the end of this chapter, you will understand the different retention methods you can use, how you can select different retention methods, and the use of the new subcommand operands.

4.1 Overview

Among the important decisions to be made when using DFSMSrmm is how to retain tape data sets and for how long. You might want to retain a given data set for a specific period of time after it is created, or retain it based on some event (for example, while the data set is cataloged), or retain it permanently. With z/OS V1R13, DFSMSrmm offers two retention methods: EXPDT and VRSEL. These methods are used for retention of your volumes and data sets on your volumes. For each volume set, you can specify whether the volumes and data sets in that volume set need to be managed by the expiration date or by VRSEL retention vital record specification (VRS) policies.

Every volume has a retention method: either EXPDT or VRSEL. The default retention method specified in parmlib is used, unless a method is assigned by the EDG_EXIT100 installation exit, by the RMM TSO ADDVOLUME or CHANGEVOLUME subcommand. When data is created, the storage administrator can select an appropriate retention method for a volume set.

4.2 Specifying a retention method

The EDGRMMxx parmlib member has a new operand to set the system-wide retention method for new tape volumes or tape volume sets created during Open/Close/End of Volume (O/C/EOV) processing, and for tape volumes added to the DFSMSrmm control data set (CDS).

4.2.1 Expiration date

One important parameter for both retention methods is the expiration date of the data set and the volume. The data set expiration date or retention period is determined when a new tape data set is created. The expiration date or retention period can be specified at multiple levels:

- The default retention period RETPD specified in the DFSMSrmm parmlib member
- The EXPDT or RETPD in the DFSMS data class if the data set is associated with a DFSMS data class
- The JCL DD statement, using the EXPDT or RETPD keywords
- The DFSMSrmm installation exit EDG_EXIT100

Note: When you change the expiration date for a data set record representing one part of a multivolume data set on volumes managed by the EXPDT retention method, DFSMSrmm updates the expiration date and time for all the data set records for the data set.

4.2.2 Retention date

z/OS V1R13 DFSMSrmm provides enhancements to make tape management easier and to improve administrator productivity by providing the following functions:

- A retention date in the volume

  With this enhancement, the retention date is displayed instead of the expiration date in the search results list, if the volume or data set is VRS-retained.
- A data set search result

When a resource is retained by VRS, the search results list for volumes or data sets might show retained resources with an expiration date that is already passed. The storage administrator can more easily determine from the search results list why the volume is retained, without viewing the volume and data set details.

4.2.3 Using the EDGRMMnn parmlib option RETENTIONMETHOD

Use this operand to set the system-wide retention method default for new tape volume sets. New tape volume sets can be created during Open/Close/End of Volume (O/C/EOV) processing, or through DFSMSrmm commands. A tape volume set can be a multi-volume set or a single tape volume. RETENTIONMETHOD can be abbreviated as RM as shown in Figure 4-1.

```
                          |—RETENTENMETHOD(VRSEL)———|
                          |                         |
▸—OPTION——————————————————————————————————————————————————————————————◂◂
                          |                         |
                          |—RETENTENMETHOD(--————)—|
                            RM              |—EXPDT—|
```

Figure 4-1 EDGRMMnn RETENTIONMETHOD operand

The following values apply:

VRSEL The VRSEL retention method can retain a volume beyond its expiration date by using a VRS to define a retention policy. The VRSEL retention method can also be used for setting a movement policy. Volumes and data sets managed by this retention method are subject to frequent VRSEL processing. In every run of VRSEL processing, matching your data sets and volumes to the vital record policies is done again, so the retention and movement management can change from one inventory management run to another. This is the default.

EXPDT The EXPDT retention method is based on the expiration date and avoids VRSEL processing. As a result, the retention information can be known when a tape data set is created. You can also manually override an expiration date to release a volume before the original expiration date is reached. The movement of the volumes must be managed manually.

Use the RMM TSO LISTCONTROL subcommand with the OPTION operand to list the current setting. Figure 4-2 on page 164 shows the result of the LISTCONTROL subcommand that includes the new retention method information.

```
System options:
PARMLIB Suffix  = 02
Operating mode  = P    Retention period: Default = 0       Maximum = NOLIMIT
                                         Catalog = 6     hours
                       Retention method: Method  = VRSEL
Control data set name      = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold          = 75%               Journal transaction = NO
Catalog SYSID              = Notset
Scratch procedure name     = EDGXPROC
Backup procedure name      = EDGCDSBK
IPL date check  = N        Date format    = J       RACF support   = N
SMF audit       = 42       SMF security   = 42      CDS id         = SC70
MAXHOLD value   = 100      Lines per page = 54      System ID      = SC70
BLP             = RMM      TVEXT purge    = RELEASE Notify         = N
                                 days     = 0
Uncatalog       = Y        VRS job name   = 2       Message case   = M
MASTER overwrite= USER     Accounting     = J       VRS selection  = NEW
VRS change      = INFO     GDG duplicate  = BUMP    GDG cycle by = GENERATION
VRSMIN action   = INFO     VRSMIN count   = 1
VRSDROP action  = INFO     VRSDROP count  = 0              percent = 10
VRSRETAIN action= INFO     VRSRETAIN count= 0              percent = 80
EXPDTDROP action= INFO     EXPDTDROP count= 0              percent = 10
Disp DD name    = DISPDD   Disp msg ID    = EDG4054I
Retain by       = SET      Move by        = SET     CMDAUTH Owner  = NO
PREACS          = NO       SMSACS         = NO      CMDAUTH Dsn    = YES
Reuse bin       = CONFIRMMOVE                       Media name     = 3480
                                                    Local tasks    = 10
PDA: ON
 Block count    = 0        Block size     = 0       Log            = ON
SMSTAPE:
 Update scratch = YES      Update command = YES     Update exits   = YES
 Purge          = ASIS
Client/Server:
 Subsystem type = STANDARD Port           = 0
 Server                    Server tasks   = 0
  host name  =
  IP address =
```

Figure 4-2 LISTCONTROL OPTION output

4.2.4 EDG_EXIT100 retention method support

You can use the EDG_EXIT100 installation exit to set the retention method to be used for a new tape volume or tape volume set. When you create a new tape volume or tape volume set, or rewrite an existing set from the first file, you can override the system default retention method.

With z/OS V1R13, every volume has a retention method, either EXPDT or VRSEL. The default retention method that is specified in parmlib is used, unless one is assigned by the EDG_EXIT100 installation exit or by the ADDVOLUME or CHANGEVOLUME command. The tape copy application can use the EDGPL100 macro to invoke the installation exit EDG_EXIT100 to copy the data set attributes from the source data set to the copy data set during OPEN processing. EDG_EXIT100 can notify DFSMSrmm that the data set being created is being copied from another. During OPEN processing, the exit can identify the source data set from which DFSMSrmm will obtain all existing data set attributes, which will be used for the target data set. DFSMSrmm EOV processing ensures that the attributes are copied to all target data set records when the output data set becomes a multivolume data set.

Tailor the sample EDGUX100 exit module

Update the sample EDGUX100 exit module based on your retention requirements. You must perform these tasks:

1. Define the system default retention method. See the PARMLIB Member EDGRMMxx OPTION command.
2. Define the DFSMSrmm default and maximum retention periods with OPTION RETPD and MAXRETPD.
3. Update the sample EDGUX100 exit module based on your retention requirements:
   a. Copy the sample EDGUX100 exit module and use the copy as a base for your exit module.
   b. Update the exit module. Perform your processing only when the PL100_CAN_RETMET bit is set to B'1'. Set PL100_RETENTIONMETHOD to the value required, and ensure the PL100_SET_RETMET bit is set to B'1'. If you do not set a retention method, the system default retention method is used.
   c. Make any other changes required, such as setting or clearing the EXPDT. The sample EDGUX100 exit module includes an example of setting the retention method. To use the sample EDGUX100 exit module for this function, modify the table as shown in Figure 4-3 on page 166. The order in which the table entries are listed is important because the exit scans the table until it finds the first entry where the job name, data set name, and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

The sample EDGUX100 exit module includes an example of setting the retention method. To use the sample EDGUX100 exit module for this function, modify the table as shown in Figure 4-3 on page 166. The order in which the table entries are listed is important because the exit scans the table until it finds the first entry where the job name, data set name, and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

```
RMTAB  DS OF                          START OF RM TABLE
       SPACE 1
       DC    CL8'*'                   JOBNAME
       DC    CL44'RMMUSER.RMVRSEL.*'  DATA SET NAME
       DC    CL8'*'                   PROGRAM NAME
       DC    AL1(PL100_RM_VRSEL)      RETENTION METHOD VRSEL
       DC    XL3'00'                  RESERVED
       SPACE 1
       DC    CL8'*'                   JOBNAME
       DC    CL44'RMMUSER.RMEXPDT.*'  DATA SET NAME
       DC    CL8'*'                   PROGRAM NAME
       DC    AL1(PL100_RM_EXPDT)      RETENTION METHOD EXPDT
       DC    XL3'00'                  RESERVED
       SPACE 1
       DC    CL8'RM END'              END OF RM TABLE MARKER
```

Figure 4-3 Sample RETENTIONMETHOD selection table

The following values apply to Figure 4-3:

JOBNAME One-to-eight alphanumeric or national characters including % and *.
% can be used to ignore a positional character in the job name.
* can be used to ignore all remaining characters in the job name.
A jobname of * means that the entry applies to all jobs.

DATA SET NAME Can be up to forty-four characters, following z/OS data set naming conventions, including % and *.
% can be used to ignore a positional character in the data set name.
***** can be used to ignore all remaining characters in the data set name.
A data set name of * means that the entry applies to all data sets.

The use of the character * is not the same as in the generic data set names supported by DFSMSrmm for vital records specifications and search data set masks. Here, the * works like the characters *.* might in a generic data set name mask.

PROGRAM NAME A value up to eight alphanumeric character including % and *.
% can be used to ignore a positional character in the program name.
* can be used to ignore all remaining characters in the program name.
A program name of * means that the entry applies to all programs.

retention method Select one of the following methods:

PL100_RM_VRSEL To assign the retention method VRSEL
PL100_RM_EXPDT To assign the retention method EXPDT

To compile and link the updated EDGUX100 user exit, use the JCL shown in Figure 4-4 on page 167.

```
//ASMCL    PROC SM=,LM=
//C        EXEC PGM=ASMA90,PARM='NODECK,XREF(SHORT),LINECOUNT(58)'
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(7040,400)
//SYSUT2   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSUT3   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSIN    DD   DSN=RMM.ADDONS.SOURCE(&SM),DISP=SHR
//L        EXEC PGM=HEWL,PARM='LIST,MAP,XREF,RENT,REUS,',COND=(4,LT,C)
//SYSLMOD  DD   DSN=SYS1.SANDBOX.MIGLIB(&LM),DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//         PEND
//COMPLNKO EXEC ASMCL,LM=EDGUX100,SM=EDGUX100
```

Figure 4-4 Compile and link updated EDGUX100 user exit

You can use the SETPROG EXIT command to control exits that have been defined to the dynamic exits facility. To activate or change the current used EDGUX100 user exit, you have to use the command shown in Figure 4-5. This command replaces the old sequence that was required before z/OS V1.12.

```
SETPROG EXIT,REFRESH,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 4-5 SETPROG command syntax to refresh a dynamic exit

For versions before z/OS V1.12, the correct sequence to replace an existing exit is shown in Figure 4-6.

```
MODIFY DFRMM,QUIESCE

SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

MODIFY DFRMM,M=nn
```

Figure 4-6 SETPROG command syntax sequence before z/OS V1.12

The following values apply:

ADD Adds an exit routine to an exit
REPLACE Replaces an exit routine for an exit
DELETE Deletes an exit routine from an exit
EXITNAME The 1 - 16 character name of the exit
MODNAME The 1 - 8 character name of the exit routine. If DSNAME is not specified, the system tries to locate the exit routine using the link pack area (LPA), the LNKLST concatenation, and the nucleus.

Tip: DFSMSrmm provides a second sample for EDGUX100. This sample is called EDGCVRSX. It is different from the EDGUX100 sample because the special date, retention method, VRSELEXCLUDE, and pooling function are table driven and you can change the table dynamically. For documentation about using EDGCVRSX for EDGUX100, see SAMPLIB member EDGCMM01.

4.2.5 Subcommands for RETENTIONMETHOD parameters

You can use the TSO RMM ADDVOLUME or CHANGEVOLUME subcommands with the RETENTIONMETHOD operand to set the retention method for a tape volume or tape volume set. Specify this operand for the first volume in a multivolume sequence. All other volumes that are added to the set assume the same retention method.

After a retention method is defined for a non-scratch volume, it is not overridden to the system-wide default during OPEN output processing, but can be changed by installation exit EDG_EXIT100. Volumes in a set always assume the retention method of the first volume in the set.

Specify EXPDT to set the retention method for a tape volume set to be based on EXPDT. Data sets and volumes managed by this retention method are never processed by VRSEL inventory management.

Specify VRSEL to set the retention method for a tape volume set to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to VRSs, and if a match is found, to determine whether the data set or volumes are to be retained by VRS.

4.2.6 Syntax format of the RETENTIONMETHOD operand

Use this operand to set the retention method for a tape volume set. Specify this operand for the first volume in a multivolume sequence. All other volumes added to the set assume the same retention method. Figure 4-7 on page 169 shows the correct syntax for a command that can be used to set the retention method to EXPDT or VRSEL.

Important: Authorization is based on the following access:

- STGADMIN.EDG.MASTER access if the following resource is not defined
- UPDATE access to STGADMIN.EDG.CV.RM to allow any volume to be updated

```
                         |                  |-parmlib_option-|   |
                         |--RETENTIONMETHOD(--------------------)-|
                         |                                       |
>>-- ADDVOLUME- VOLSER ------------------------------------------------------><<
    AV                   |                                       |
                         |--RETENTIONMETHOD(--------------------)-|
                            RM              |-EXPDT----------|
                                            |-VRSEL----------|


                           |                  |-parmlib_option-|   |
                           |--RETENTIONMETHOD(--------------------)-|
                           |                                       |
>>-- CHANGEVOLUME- VOLSER ------------------------------------------------------><<
    CV                     |                                       |
                           |                  |-parmlib_option-|   |
                           |--RETENTIONMETHOD(--------------------)-|
                              RM              |-EXPDT----------|
                                              |-VRSEL----------|
```

Figure 4-7 RETENTIONMETHOD operand command syntax

The following values apply:

EXPDT To set the retention method for a tape volume set to be based on EXPDT. Data sets and volumes that are managed by this retention method are never processed by VRSEL inventory management.

VRSEL To set the retention method for a tape volume set to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to VRSs, and if a match is found, to determine whether the data set or volumes are to be retained by VRS.

Note: When you manually define or change a volume that is a start of a volume set, you can specify the retention method for the volume set. All other volumes that are added to the set assume the same retention method. If you do not specify a retention method, DFSMSrmm will use the default retention method that is specified by the RETENTIONMETHOD option in EDGRMMxx.

4.2.7 Using the SEARCHVOLUME subcommand

You can use the SEARCHVOLUME subcommand with the RETENTIONMETHOD operand to specify a list limited to volumes that have a specified retention method. Specify EXPDT to select volumes with the EXPDT retention method. Specify VRSEL to select volumes with the VRSEL retention method. The new RETENTIONMETHOD operand syntax is shown in Figure 4-8 on page 170.

```
▶▶── SEARCHVOLUME─ VOLUME(VOLSER) ──────────────────────────────────────────◀◀
     SV                                  │                                  │
                                         │──RETENTIONMETHOD(──────────────)─│
                                            RM                 │─EXPDT─│
                                                               │─VRSEL─│
```

Figure 4-8 RETENTIONMETHOD operand command syntax

The following values apply:

EXPDT Sets the retention method for a tape volume set to be based on EXPDT. Data sets and volumes managed by this retention method are never processed by VRSEL inventory management.

VRSEL Sets the retention method for a tape volume set to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to VRSs, and if a match is found, to determine whether the data set or volumes are to be retained by VRS.

Example one

Figure 4-9 shows how you can use the RMM SEARCHVOLUME subcommand to get a list of all volumes owned by user SCHLUM.

Note: If you do not specify the RETENTIONMETHOD operand, DFSMSrmm searches all volumes, regardless of the retention method assigned to a volume.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM)
```

Figure 4-9 SEARCHVOLUME without the RETENTIONMETHOD operand

You get the list of all six volumes that are currently owned by SCHLUM, as shown in Figure 4-10.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0008 SCHLUM         2011/324   2011/333   VTFM001  1     M
VT0013 SCHLUM         2011/324   2011/333   VTFM001  1     M
VT0015 SCHLUM         2011/324   2011/333   VTFM001  1     M
VT0016 SCHLUM         2011/324   2011/333   VTFM001  1     M
VT0017 SCHLUM         2011/325   2011/334   VTFM001  1     M
VT0021 SCHLUM         2011/325   2011/334   VTFM001  1     M
EDG3012I 6          ENTRIES LISTED
```

Figure 4-10 SEARCHVOLUME result without using the RETENTIONMETHOD operand

Example two

Figure 4-11 on page 171 shows how you can use the RMM SEARCHVOLUME subcommand with the RETENTENMETHOD(EXPDT) operand to get only a list of all volumes owned by user SCHLUM but that have a retention method of EXPDT.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM) RETENTIONMETHOD(EXPDT)
```

Figure 4-11 SEARCHVOLUME using the RETENTIONMETHOD(EXPDT) operand

You get the list of volumes that are assigned to owner SCHLUM and have a retention method of EXPDT set in the volume record as shown in Figure 4-12.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0008 SCHLUM         2011/324   2011/333   VTFM001  1     M
VT0013 SCHLUM         2011/324   2011/333   VTFM001  1     M
EDG3012I 2          ENTRIES LISTED
```

Figure 4-12 SEARCHVOLUME result using the RETENTIONMETHOD(EXPDT) operand

Now, you can use the LISTVOLUME subcommand to get all the details of one of the listed volumes as shown in Figure 4-13.

```
RMM LISTVOLUME VT0008
```

Figure 4-13 LISTVOLUME subcommand

Example 4-1 shows the result of the LISTVOLUME subcommand.

Example 4-1 LISTVOLUME VT008 result

```
Volume information:
Volume = VT0008   VOL1  =          Rack  =           Owner   = SCHLUM
  Type = LOGICAL          Stacked count  = 0         Jobname = TEST9999
  Worldwide ID =                                     WORM    = N
Creation:  Date = 2009/222    Time = 12:49:45  System ID = SC70
Assign:    Date = 2011/324    Time = 16:22:36  System ID = SC70
                                               User ID   = SCHLUM
Expiration date = 2011/333   Original      = 2011/333
         set by = OCE_JFCB
Retention date  =            Set retained  = NO
Retention method= EXPDT
         set by = OCE_DEF
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume status:             Hold = N   File 1 Data set seq = 1
Status = MASTER    Availability =                   Label = SL
Current label version =            Required label version =
Media information: VTFM
Density = IDRC  Type = VT3590G2  Format = VT3590G2  Compaction = YES
Special attributes   = NONE      Vendor =
Encryption Key Labels:                                          Method:
1=
2=
Action on release:
Scratch immediate  = N  Expiry date ignore = N
Scratch = Y  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Actions pending:
Scratch = N  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Storage group  =
Loan location  =           Account = 999,POK
Old loan loc   =
```

```
Description    =
Security class =           Description =
```

The following values apply:

Retention method The Retention method field specifies which retention processing to use. The retention method can only be changed for the first volume in a multi-volume sequence and will be assumed for all volumes of the set. The retention method cannot be changed for a SCRATCH volume unless you also specify the STATUS field. The following value is valid:

- **EXPDT** The retention method for a tape volume set is only based on the expiration date. Data sets and volumes that are managed by this retention method are never processed by VRSEL inventory management.

- **set by** This field shows where the retention method is from:
  - **UNDEFINED** Not set.
  - **CMD** Retention method set by TSO subcommand.
  - **CMD_DEF** Default retention method applied during subcommand processing.
  - **OCE_DEF** Default retention method applied during tape recording.
  - **OCE_EXIT** EDG_EXIT100 set retention method during tape recording.
  - **LCS_DEF** Default retention method applied for system-managed tapes when called from object access method (OAM) installation exits.
  - **CNVT** Retention method set during conversion.
  - **EXPORT_DEF** Default retention method applied during export processing.
  - **INERS_DE** Default retention method applied during tape initialization.

Example three

Figure 4-14 shows how to use the RMM SEARCHVOLUME subcommand with the RETENTIONMETHOD(VRSEL) operand to get a list of all volumes owned by user SCHLUM that have a retention method of `VRSEL`.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM) RETENTIONMETHOD(VRSEL)
```

Figure 4-14 SEARCHVOLUME using the RETENTIONMETHOD(VRSEL) operand

Figure 4-15 shows a list of volumes assigned to owner SCHLUM that have a retention method of VRSEL set in the volume record.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0015 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0016 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0017 SCHLUM          2011/325   2011/334   VTFM001  1     M
VT0021 SCHLUM          2011/325   2011/334   VTFM001  1     M                SCHLUM
EDG3012I 4          ENTRIES LISTED
```

Figure 4-15 SEARCHVOLUME result using the RETENTIONMETHOD(VRSEL) operand

Now you can use the LISTVOLUME subcommand to get all the details of one of the listed volumes, as shown in Figure 4-16.

```
RMM LISTVOLUME VT0021
```

Figure 4-16 LISTVOLUME subcommand

Figure 4-17 shows the result of the LISTVOLUME subcommand.

```
Volume information:
Volume = VT0021   VOL1  =          Rack  =           Owner   = SCHLUM
  Type = LOGICAL          Stacked count  = 0         Jobname = TEST9999
  Worldwide ID =                                     WORM    = N
Creation:  Date = 2009/222    Time = 12:49:45  System ID = SC70
Assign:    Date = 2011/325    Time = 10:59:17  System ID = SC70
                                               User ID   = SCHLUM
Expiration date = 2011/334   Original      = 2011/334
         set by = OCE_JFCB
Retention date  =            Set retained  = NO
Retention method= VRSEL
         set by = OCE_DEF
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume status:             Hold = N   File 1 Data set seq = 1
Status = MASTER    Availability =                   Label = SL
Current label version =            Required label version =
Media information: VTFM
Density = IDRC  Type = VT3590G2  Format = VT3590G2  Compaction = YES
Special attributes   = NONE      Vendor =
Encryption Key Labels:                                          Method:
1=
2=
Action on release:
Scratch immediate  = N  Expiry date ignore = N
Scratch = Y  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Actions pending:
Scratch = N  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Storage group  =
Loan location  =           Account = 999,POK
Old loan loc   =
Description    =
Security class =           Description =
```

Figure 4-17 LISTVOLUME VT008 result

The following values apply:

Retention method The Retention method field specifies which retention processing is to be used. The retention method can only be changed for the first volume in a multi-volume sequence. The value of this field is used for all volumes of the set. The retention method cannot be changed for a SCRATCH volume unless you also specify the STATUS field. The following value is valid:

VRSEL DFSMSrmm inventory management attempts to match data sets and volumes to VRSs, and if a match is found, to determine if the data sets or volumes are to be retained by VRS.

set by This field shows where the retention method is from:

- **UNDEFINED** Not set.
- **CMD** Retention method that is set by TSO subcommand.
- **CMD_DEF** Default retention method that is applied during subcommand processing.
- **OCE_DEF** Default retention method that is applied during tape recording.
- **OCE_EXIT** EDG_EXIT100 sets the retention method during tape recording.
- **LCS_DEF** Default retention method that is applied for system-managed tapes when called from OAM installation exits.
- **CNVT** Retention method that is set during conversion.
- **EXPORT_DEF** Default retention method that is applied during export processing.
- **INERS_DE** Default retention method that is applied during tape initialization.

5

Excluding data sets from VRSEL processing

This chapter explains how you can exclude specific data sets from Data Facility System Managed Storage removable media manager (DFSMSrmm) VRSEL processing as they are created or rewritten.

This chapter contains the following information:

- Overview
- Subcommands for VRSELEXCLUDE parameters
- Syntax format of the VRSELEXCLUDE operand
- Using EDG_EXIT100 to exclude data sets from VRSEL support
- Using the SEARCHDATASET subcommand
- Inventory Management VRSEL/EXPROC Processing

By the end of this chapter, you will understand how you can exclude data sets from vital record selection processing by using different exclusion methods.

5.1 Overview

You can use the EDG_EXIT100 installation exit to exclude specific data sets from DFSMSrmm VRSEL processing as they are created or rewritten. You can specify this for any data set, but DFSMSrmm ignores the request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes managed by the EXPDT retention method and is not affected by this support.

When DFSMSrmm excludes a data set from VRSEL processing, it ensures that the data set vital record attribute is reset and the retention date is set to the current date. The matching vital record specification (VRS) information is left unchanged.

5.2 Subcommands for VRSELEXCLUDE parameters

All data in the DFSMSrmm inventory is managed by dynamic VRS policies. For z/OS V1R13, DFSMSrmm provides a new operand VRSELEXCLUDE for the RMM CHANGEDATASET and SEARCHDATASET subcommands.

5.2.1 Syntax format of the VRSELEXCLUDE operand

Use this operand to override DFSMSrmm VRSEL processing. You can specify this for any data set on a volume that is managed by the VRSEL retention method. If VRSELEXCLUDE(YES) is specified for a data set that is already retained as a vital record, its vital record attribute is reset and the retention date is set to the current date. The data set VRSELEXCLUDE attribute is set to YES for all data sets on volumes that are managed by the EXPDT retention method. For more information about the EXPDT retention method, see Chapter 4, “DFSMSrmm retention methods” on page 161.

When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record. That is, set this attribute for one data set record for each of the volumes on which the data set resides.

Figure 5-1 shows the syntax that can be used to set the vital record selection processing to YES or NO.

```
▶▶── CHANGEDATASET─ dsn ─VOLUME(vol) ──────────────────────────────────────◀◀
     CD                                 │                          │
                                        │──VRSELEXCLUDE(──────────)─│
                                           VX            │─NO──│
                                                         │─YES─│
```

Figure 5-1 RETENTIONMETHOD operand command syntax

The definitions are listed:

NO Ensures a data set is included in VRSEL processing
YES Excludes a data set from VRSEL processing

Important: Authorization requires the following access:

- STGADMIN.EDG.MASTER access if the following resource is not defined
- UPDATE access to STGADMIN.EDG.CD.VX to allow any data set to be updated

5.2.2 Using EDG_EXIT100 to exclude data sets from VRSEL support

A new option is provided via EDG_EXIT100 to request to override DFSMSrmm VRSEL processing for specific data sets as they are created or rewritten. You can specify this for any data set, but DFSMSrmm ignores your request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes that are managed by the EXPDT retention method. It is not affected by this support. If a data set is already retained as a vital record, the vital record attribute is reset and the retention date is set to the current date.

When DFSMSrmm honors your request to exclude a data set from VRSEL, processing ensures that the data set vital record attribute is reset and the retention date is set to the current date. The matching VRS information is left unchanged.

You can use the EDG_EXIT100 installation exit to set the VRSELEXCLUDE attribute for VRSEL-managed data sets at OPEN time when you create a new tape data set or rewrite existing data sets.

When DFSMSrmm honors your request to exclude a data set from VRSEL, processing ensures that the data set vital record attribute is reset and the retention date is set to the current date. The matching VRS information is left unchanged.

You can use the EDG_EXIT100 installation exit to set the VRSELEXCLUDE attribute for VRSEL-managed data sets at OPEN time when you create a new tape data set or rewrite existing data sets.

Update the sample EDGUX100 exit module based on your retention requirements. You must perform these tasks:

1. Copy the sample EDGUX100 exit module and use the copy as a base for your exit module.
   a. Update the exit module. Only perform your processing when the PL100_CAN_VRSELEXCLUDE bit is set to B'1'.
   b. Set the PL100_SET_VRSELEXCLUDE bit to B'1' for data sets. If you do not request VRSELEXCLUDE, the default value for the retention method will be used. If the installation exit sets PL100_SET_VRSELEXCLUDE, any VRS management value set in PL100_VRS is ignored.
   c. You do not need to set the PL100_SET_VRSELEXCLUDE bit when you also request to set the retention method to EXPDT. DFSMSrmm always sets the VRSELEXCLUDE attribute for data sets that are managed by the EXPDT retention method.
2. Make any other changes required, such as setting the retention method when creating the first file on the tape or clearing the EXPDT retention method.

The sample EDGUX100 exit module includes an example of setting the VRSELEXCLUDE attribute. To use the sample EDGUX100 exit module for this function, modify the table as shown in Figure 5-2 on page 178. The order in which the table entries are listed is important because the exit scans the table until it finds the first entry where the job name, data set name, and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

```
VXTAB DS 0F                        START OF VRSELEXCLUDE TABLE
      SPACE 1
      DC    CL8'*'                 JOBNAME
      DC    CL44'RMMUSER.VX.*'     DATA SET NAME
      DC    CL8'*'                 PROGRAM NAME
      DC    CL8'SCHLUM*'           JOBNAME
      DC    CL44'RMM.ADDONS.*'     DATA SET NAME
      DC    CL8'*'                 PROGRAM NAME
      DC    CL8'BACK%%%%'          JOBNAME
      DC    CL44'SSC.BACK*'        DATA SET NAME
      DC    CL8'ADRDSSU'           PROGRAM NAME
      SPACE 1
      DC    CL8'VX END'            END OF VX TABLE MARKER
```

Figure 5-2 Sample VRSELEXCLUDE selection table

The following definitions relate to Figure 5-2:

JOBNAME One-to-eight alphanumeric or national characters that include the percent sign (%) and the asterisk (*). The percent sign (%) can be used to ignore a positional character in the job name. The asterisk (*) can be used to ignore all remaining characters in the job name. A jobname of an asterisk (*) means that the entry applies to all jobs.

DATA SET NAME Can be up to forty-four characters, following z/OS data set naming conventions, including percent sign (%) and the asterisk (*). The percent sign (%) can be used to ignore a positional character in the data set name. The asterisk (*) can be used to ignore all remaining characters in the data set name. A data set name of an asterisk (*) means that the entry applies to all data sets.

The use of the character asterisk (*) is not the same as in the generic data set names that are supported by DFSMSrmm for vital records specifications and search data set masks. Here, the asterisk (*) works like the characters *.* might work in a generic data set name mask.

PROGRAM NAME A value up to eight alphanumeric characters that include the percent sign (%) and the asterisk (*). The percent sign (%) can be used to ignore a positional character in the program name. The asterisk (*) can be used to ignore all remaining characters in the program name. A program name of an asterisk (*) means that the entry applies to all programs.

To compile and link the updated EDGUX100 user exit, use the JCL shown in Figure 5-3 on page 179.

```
//ASMCL    PROC SM=,LM=
//C        EXEC PGM=ASMA90,PARM='NODECK,XREF(SHORT),LINECOUNT(58)'
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(7040,400)
//SYSUT2   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSUT3   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSIN    DD   DSN=RMM.ADDONS.SOURCE(&SM),DISP=SHR
//L        EXEC PGM=HEWL,PARM='LIST,MAP,XREF,RENT,REUS,',COND=(4,LT,C)
//SYSLMOD  DD   DSN=SYS1.SANDBOX.MIGLIB(&LM),DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//         PEND
//COMPLNKO EXEC ASMCL,LM=EDGUX100,SM=EDGUX100
```

Figure 5-3 Compile and link the updated EDGUX100 user exit

To activate or change the currently used EDGUX100 user exit, use the commands shown in Figure 5-4. If the exit was loaded previously, you must delete the commands and then use the add function.

```
SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 5-4 SETPROG command syntax

Figure 5-5 shows the correct sequence to replace an existing exit for z/OS versions before V1.12.

```
MODIFY DFRMM,QUIESCE
SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
MODIFY DFRMM,M=nn
```

Figure 5-5 SETPROG command syntax sequence before z/OS V1.12

The following definitions apply to Figure 5-5:

ADD Adds an exit routine to an exit.
REPLACE Replaces an exit routine for an exit.
DELETE Deletes an exit routine from an exit.
EXITNAME The 1-16 character name of the exit.
MODNAME The 1-8 character name of the exit routine. If DSNAME is not specified, the system tries to locate the exit routine using the LPA, the LNKLST concatenation, and the nucleus.

Tip: DFSMSrmm provides a second sample for EDGUX100. This sample is called EDGCVRSX. It is different from the EDGUX100 sample because the special date, retention method, VRSELEXCLUDE, and pooling function are table driven and you can change the table dynamically. For information about EDGCVRSX for EDGUX100, see SAMPLIB member EDGCMM01.

5.2.3 Using the SEARCHDATASET subcommand

Use the SEARCHVOLUME subcommand with the VRSELEXCLUDE operand to specify a list that is limited to data sets that have a specified vital record exclude method. Specify YES to exclude a data set from VRSEL processing. Specify NO to ensure that a data set is included in VRSEL processing. The new VRSELEXCLUDE operand syntax is shown in Figure 5-6.

```
                                          |-VOLUME(*)-----------|
                   |-DSNAME(*)--------|   |-OWNER(cmd_issuer_id)-|
                   |                  |   |                     |
>>-- SEARCHDATASET------------------------------------------------------------>
     SD            |                  |   |                     |
                   |-DSNAME(dsn_mask)-|   |-VOLUME(VOLSER)------|
                     DS                   | VO                  |
                                          |-OWNER(owner_id)-----|
                                            OW
   |-FILESEQ(1)----------------|
   |                           |
>------------------------------------------------------------------------------>
   |                           |   |                           |
   |-FILESEQ(physical_file_seq)-|   |--VRSELEXCLUDE(---------)-|
     SEQ                           VX            |-NO--|
                                                 |-YES-|
```

Figure 5-6 VRSELEXCLUDE operand command syntax

The explanations are listed:

NO	Ensures that a data set is included in VRSEL processing
YES	Excludes a data set from VRSEL processing

Example one

Figure 5-7 shows how you can use the RMM SEARCHDATASET subcommand to get a list of all data sets that are owned by the user SCHLUM.

> **Note:** If you do not specify the VRSELEXCLUDE operand, DFSMSrmm searches all data sets, regardless of the vital record selection criteria set for a data set.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM)
```

Figure 5-7 SEARCHDATASET without the VRSELEXCLUDE operand

You get the list of all 10 data sets that currently have the owner SCHLUM, as shown in Figure 5-8 on page 181.

```
Data set name                                  Volume Owner    Create date Seq
---------------------------------------------- ------ -------- ----------- -----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0008 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0013 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0015 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0016 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0017 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0021 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0022 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0023 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0024 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0025 SCHLUM   2011/325    1
EDG3012I 10         ENTRIES LISTED
```

Figure 5-8 SEARCHDATASET result without using the VRSELEXCLUDE operand

Example two

Figure 5-9 shows how you can use the RMM SEARCHDATASET subcommand to get a list of all data sets owned by user SCHLUM and have VRSELEXCLUDE set to YES.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM) VRSELEXCLUDE(YES)
```

Figure 5-9 SEARCHDATASET using VRSELEXCLUDE(YES) operand

Figure 5-10 shows that only these eight data sets have VRSELEXCLUDE set to YES.

```
Data set name                                  Volume Owner    Create date Seq
---------------------------------------------- ------ -------- ----------- ----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0008 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0013 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0017 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0021 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0022 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0023 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0024 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG            VT0025 SCHLUM   2011/325    1
EDG3012I 8          ENTRIES LISTED
```

Figure 5-10 SEARCHDATASET using VRSELEXCLUDE(YES) operand

Now, you can use the LISTDATASET subcommand to get all the details of one of the listed data sets as shown in Figure 5-11.

```
RMM LD 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG' VOL(VT0025)
```

Figure 5-11 LISTDATASET subcommand

Example 5-1 shows the result of the LISTDATASET subcommand.

Example 5-1 LISTDATASET result

```
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume        = VT0025                Physical file sequence number = 1
Owner         = SCHLUM                            Data set sequence = 1
Create date   = 2011/325   Create time = 19:35:33 System ID         = SC70
Expiration date       = 2011/325   Original expir. date  =
```

```
         set by       = OCE_DEF
Block size            = 80         Block count           = 1
Data set size(KB)     = 1
Physical size(KB)     = 0          Compression           = 0.00
Percent of volume     = 0          Total block count     = 2
Logical Record Length = 0          Record Format         = F
Date last written     = 2011/325   Date last read        = 2011/325
Job name              = TEST9999   Last job name         = TEST9999
Step name             = STEP03     Last step name        = STEP03
Program name          = EOVTEST    Last program name     = EOVTEST
DD name               = OUT        Last DD name          = OUT
Device number         = 0406       Last Device number    = 0406
Management class      =            VRS management value  =
Storage group         =            VRS retention date    =
Storage class         =            VRS retained          = NO
Data class            =            Closed by Abend       = NO
                                   Deleted               = NO
VRSEL exclude         = YES        Catalog status        = UNKNOWN
Primary VRS details:
       Name           =
       Job name       =            Type                  =
       Subchain NAME  =            Subchain start date   =
Secondary VRS details:
       Value or class =
       Job name       =
       Subchain NAME  =            Subchain start date   =
Security Class   =            Description   =
BES key index    = 0

Last Change information:
Date             = 2011/325    Time = 20:25:03  System  = SC70
User change date = 2011/325    Time = 20:25:03  User ID = SCHLUM
```

The following list describes Example 5-1 on page 181:

VRS retention date This field displays the date by which a data set is no longer retained by the current VRS. For data sets not retained by a VRS, the retention date is either the date no longer retained by VRS, or null. The following values are valid:

- **Calendar date** The date that is calculated by VRSEL processing in the currently selected date format
- **WHILECATLG** Data set is retained by a VRS with WHILECATALOG
- **CYCL/*ccccc*** Data set is retained by a VRS defined with CYCLES, where *ccccc* is the COUNT of cycles
- **CATRETPD** Data set is retained by a VRS defined with WHILECATALOG and the data set is not cataloged, but the CATRETPD parmlib option applies
- **PERMANENT** Data set is retained by a VRS defined with DAYS COUNT(99999)

VRSEL exclude This field specifies whether the data set is excluded from vital record processing. The VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method. It can optionally be set to YES for any data set on a volume that is managed by the VRSEL retention method. When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record - one data set record for each of the volumes on which the data set resides. The following values are valid:

YES Exclude a data set from VRSEL processing
NO Ensure a data set is included in VRSEL processing

Note: If a data set is not vital record selected, the retention date field is always empty.

Example three

Figure 5-12 shows how you can use the RMM SEARCHDATASET subcommand to get a list of all data sets that are owned by user SCHLUM and have the VRSELEXCLUDE attribute set to NO.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM) VRSELEXCLUDE(NO)
```

Figure 5-12 SEARCHDATASET using VRSELEXCLUDE(NO) operand

Figure 5-13 shows that only these two data sets have the VRSELEXCLUDE attribute set to NO.

```
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- ----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0015 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0016 SCHLUM   2011/324    1
EDG3012I 2          ENTRIES LISTED
```

Figure 5-13 SEARCHDATASET using VRSELEXCLUDE(NO) operand

Now, you can use the LISTDATASET subcommand to get all the details for one of the listed data sets as shown in Figure 5-14.

```
RMM LD 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG' VOL(VT0015)
```

Figure 5-14 LISTDATASET subcommand

Example 5-2 shows the result of the LISTDATASET subcommand.

Example 5-2 LISTDATASET result

```
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume        = VT0015                Physical file sequence number = 1
Owner         = SCHLUM                             Data set sequence = 1
Create date   = 2011/324   Create time = 19:41:58 System ID         = SC70
Expiration date       = 2011/333   Original expir. date  = 2011/333
         set by       = OCE_JFCB
Block size            = 80         Block count           = 1
Data set size(KB)     = 1
Physical size(KB)     = 0          Compression           = 0.00
Percent of volume     = 0          Total block count     = 1
Logical Record Length = 0          Record Format         = F
```

```
Date last written     = 2011/324   Date last read        = 2011/324
Job name              = TEST9999   Last job name         = TEST9999
Step name             = STEP03     Last step name        = STEP03
Program name          = EOVTEST    Last program name     = EOVTEST
DD name               = OUT        Last DD name          = OUT
Device number         = 044C       Last Device number    = 044C
Management class      =            VRS management value  =
Storage group         =            VRS retention date    = PERMANENT
Storage class         =            VRS retained          = YES
Data class            =            Closed by Abend       = NO
                                   Deleted               = NO
VRSEL exclude         = NO         Catalog status        = UNKNOWN
Primary VRS details:
       Name           = RMM.F13002.**
       Job name       =            Type                  = DATASET
       Subchain NAME  =            Subchain start date   =
Secondary VRS details:
       Value or class =
       Job name       =
       Subchain NAME  =            Subchain start date   =
Security Class   =            Description   =
BES key index    = 0

Last Change information:
Date             = 2011/325    Time = 20:25:03  System  = SC70
User change date = 2011/325    Time = 20:25:03  User ID = SCHLUM
```

The following explanations apply to Example 5-2 on page 183:

Retention Date This field displays the date by which a data set is no longer retained by the current VRS. For data sets not retained by a VRS, the retention date is either the date no longer retained by VRS, or null. The following values are valid:

- **Calendar date** The date calculated by VRSEL processing in the currently selected date format
- **WHILECATLG** Data set is retained by a VRS with WHILECATALOG
- **CYCL/*ccccc*** Data set is retained by a VRS defined with CYCLES where *ccccc* is the COUNT of cycles
- **CATRETPD** Data set is retained by a VRS defined with WHILECATALOG. The data set is not cataloged, but the CATRETPD parmlib option applies
- **PERMANENT** Data set is retained by a VRS defined with DAYS COUNT(99999)

VRSEL exclude This field specifies whether the data set is excluded from vital record processing. The VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method. It can optionally be set to YES for any data set on a volume that is managed by the VRSEL retention method. When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record. Set this for one data set record for each of the volumes on which the data set resides. The following values are valid:

- **YES** Exclude a data set from VRSEL processing
- **NO** Ensure a data set is included in VRSEL processing

5.2.4 Inventory Management VRSEL/EXPROC Processing

In z/OS V1.13, the inventory management processing has changed:

- You do not need to run VRSEL processing unless volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle the expiration of all volumes managed by the EXPDT retention method.
- EXPROC processing provides a summary of volumes by retention method.
- The expiration date of volumes is set during OPEN processing, so for volumes that are managed by the EXPDT retention method, no special considerations exist for open data sets. They are managed based on the volume EXPDT.
- For volumes that are managed by the EXPDT retention method, no special considerations exist for data sets that are closed by ABEND processing or that are DELETED; they are managed based on the volume EXPDT.
- Volumes that are managed by the EXPDT retention method are included only in the EXPDTDROP limit. VRSRETAIN and VRSDROP limits apply only to volumes that are managed by the VRSEL retention method.

Updated EDGHSKP messages

For inventory management (VRSEL), there is one new message, EDG2246I. For expiration processing (EXPROC), no new messages were created. However, existing messages are expanded and adapted with new or additional information, as shown in Example 5-3.

Example 5-3 EDGHSKP messages

```
EDG6001I INVENTORY MANAGEMENT STARTING ON 2011/326 AT 13:26:39 -
               PARAMETERS IN USE ARE
         DATEFORM(J),VRSEL,EXPROC,BACKUP(AMS)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
         VRSEL(NEW)
         VRSJOBNAME(2)
         VRSMIN(1,INFO)
         VRSCHANGE(INFO)
         VRSDROP(PERCENT(10),INFO) VRSRETAIN(PERCENT(80),INFO)
               EXPDTDROP(PERCENT(10),INFO)
         SMSTAPE(PURGE(ASIS) UPDATE(EXITS,SCRATCH,COMMAND))
         CATRETPD(6)
         UNCATALOG(Y)
         TPRACF(N)
         NOTIFY(N)
         SYSID(SC64)
         CATSYSID()
         RETAINBY(SET)
         MOVEBY(SET)
         GDG(CYCLEBY(GENERATION),DUPLICATE(BUMP))
EDG2229I NUMBER OF VRS RECORDS READ IS 3
EDG2238I NUMBER OF UNUSED VRS RECORDS IS 1
EDG2246I NUMBER OF DATA SET RECORDS EXCLUDED FROM VRSEL =          8   7%
EDG2242I INITIAL NUMBER OF VRS RETAINED VOLUMES         =        107  33%
EDG2244I NUMBER OF VRS RETAINED VOLUMES TO BE DROPPED   =          0   0%
EDG2243I INITIAL NUMBER OF NEWLY ASSIGNED VOLUMES       =          2   1%
EDG2245I NUMBER OF NEWLY ASSIGNED VOLUMES TO BE RETAINED=          0   0%
EDG2444I EXIT PROCESSING DISABLED FOR THIS EXPROC RUN -
               NO ACTIVE EXIT MODULE FOR EXIT EDG_EXIT200
EDG2427I INITIAL NUMBER OF EXPDT RETAINED VOLUMES       =          6   2%
EDG2428I NUMBER OF EXPDT RETAINED VOLUMES TO BE DROPPED =          2  33%
EDG2420I PHYSICAL VOLUMES READ                          =         30   9%
```

```
EDG2420I LOGICAL  VOLUMES READ                          =        300  91%
EDG2420I RM_VRSEL VOLUMES READ                          =        322  98%
EDG2420I RM_EXPDT VOLUMES READ                          =          8   2%
EDG2420I TOTAL    VOLUMES READ                          =        330 100%
EDG2421I PHYSICAL VOLUMES UPDATED                       =          4  13%
EDG2421I LOGICAL  VOLUMES UPDATED                       =          4   1%
EDG2421I RM_VRSEL VOLUMES UPDATED                       =          6   2%
EDG2421I RM_EXPDT VOLUMES UPDATED                       =          2  25%
EDG2421I TOTAL    VOLUMES UPDATED                       =          8   2%
EDG2435I PHYSICAL VOLUMES SELECTED FOR EXPROC           =         30 100%
EDG2435I LOGICAL  VOLUMES SELECTED FOR EXPROC           =        300 100%
EDG2435I RM_VRSEL VOLUMES SELECTED FOR EXPROC           =        322 100%
EDG2435I RM_EXPDT VOLUMES SELECTED FOR EXPROC           =          8 100%
EDG2435I TOTAL    VOLUMES SELECTED FOR EXPROC           =        330 100%
EDG2424I LOGICAL  VOLUMES SET PENDING RELEASE           =          4   1%
EDG2424I RM_VRSEL VOLUMES SET PENDING RELEASE           =          2   1%
EDG2424I RM_EXPDT VOLUMES SET PENDING RELEASE           =          2  25%
EDG2424I TOTAL    VOLUMES SET PENDING RELEASE           =          4   1%
EDG2425I PHYSICAL VOLUMES RETURNED TO SCRATCH           =          4  13%
EDG2425I LOGICAL  VOLUMES RETURNED TO SCRATCH           =          2   1%
EDG2425I RM_VRSEL VOLUMES RETURNED TO SCRATCH           =          4   1%
EDG2425I RM_EXPDT VOLUMES RETURNED TO SCRATCH           =          2  25%
EDG2425I TOTAL    VOLUMES RETURNED TO SCRATCH           =          6   2%
EDG2426I PHYSICAL VOLUMES - SCRATCH RECORDS WRITTEN     =          2  10%
EDG2426I RM_EXPDT VOLUMES - SCRATCH RECORDS WRITTEN     =          2  14%
EDG2426I TOTAL VOLUMES    - SCRATCH RECORDS WRITTEN     =          2   6%
EDG2429I MAIN INVENTORY MANAGEMENT UPDATES HAVE COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK VRSEL    COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK EXPROC   COMPLETED SUCCESSFULLY
EDG6426I CONTROL DATA SET AND JOURNAL BACKUP SUCCESSFUL
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

The following message explanations apply:

EDG2246I NUMBER OF DATA SET RECORDS EXCLUDED FROM VRSEL = number percent%

Explanation During inventory management vital record processing, DFSMSrmm counts the number of data set records to be processed. The message is issued to aid VRS management. The following values are in the message text:

number Number of data set records excluded from management by VRSEL processing

percent Percentage of the total number of data set records that are eligible for management by a VRS

System action DFSMSrmm inventory management processing continues

System programmer response
Use the values to validate your policy management expectations:

Source DFSMSrmm
Detecting Module EDGVRECM
Routing Code 11
Descriptor Code 7

EDG2420I *volume_type* VOLUMES READ = *number percent%*

Explanation DFSMSrmm issues this message when the number of volumes read is greater than zero. The following values are in the message text:

volume_type This value is the volume type and it can be any of the following types:

PHYSICAL
LOGICAL
STACKED
RM_VRSEL
RM_EXPDT
TOTAL

number This value is the number of *volume_type* volumes read during inventory management

percent This value is the percentage of total volumes read by DFSMSrmm inventory management processing

System action Processing continues
Operator response None
Source DFSMSrmm
Detecting Module EDGMUPD
Routing Code 11
Descriptor Code 7

EDG2421I *volume_type* VOLUMES UPDATED = *number percent%*

Explanation DFSMSrmm issues this message when the number of volumes updated is greater than zero. The following values are in the message text:

volume_type This is the volume type and it can be any of these types:

PHYSICAL
LOGICAL
STACKED
RM_VRSEL
RM_EXPDT
TOTAL

number Number of *volume_type* volumes that were updated in the DFSMSrmm control data set

percent Percentage of the *volume_type* volumes read by DFSMSrmm inventory management processing

System action Processing continues
Operator response None
Source DFSMSrmm
Detecting Module EDGMUPD
Routing Code 11
Descriptor Code 7

EDG2424I *volume_type* TOTAL VOLUMES, THIS RUN, SET PENDING RELEASE = *number percent%*

Explanation This message is issued for information only. The following values are in the message text:

volume_type The volume type is one of the following types:

PHYSICAL
LOGICAL
RM_VRSEL
RM_EXPDT
TOTAL

number The number of *volume_type* volumes set pending release in this run of inventory management

percent This is the percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I)

System action Processing continues
Operator response None
Source DFSMSrmm
Detecting Module EDGMUPD
Routing Code 11
Descriptor Code 7

EDG2425I *volume_type* TOTAL VOLUMES RETURNED TO SCRATCH = *number percent%*

Explanation This message is issued for information only. The following information is in the message text:

volume_type The volume type is one of the following types:
PHYSICAL
LOGICAL
RM_VRSEL
RM_EXPDT
TOTAL

number Number of *volume_type* volumes returned to scratch status after all release actions have been completed

percent Percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I)

System action Processing continues
Operator response None
Source DFSMSrmm
Detecting Module EDGMUPD
Routing Code 11
Descriptor Code 7

EDG2426I *volume_type* TOTAL NUMBER OF SCRATCH RECORDS WRITTEN = *number percent%*

Explanation DFSMSrmm issues this message to the MESSAGE file when EXPROC processing for system-managed volumes requests deferred processing of volumes to scratch. The following information is in the message text:

volume_type The volume type is one of the following types:
PHYSICAL
LOGICAL

RM_VRSEL
RM_EXPDT
TOTAL

number Number of system-managed *volume_type* volumes, which are ready to return to scratch on this system, for which a record has been written to the EDGSPLCS output file

percent Percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I)

System action DFSMSrmm inventory management processing continues

Operator response None

System programmer response
Run the EDGSPLCS utility to process the scratch statements produced by the EDGHSKP EXPROC processing for system-managed volumes

Source DFSMSrmm

Detecting Module EDGMUPD

EDG2435I *volume_type* VOLUMES SELECTED FOR EXPROC = *number percent*%

Explanation DFSMSrmm issues this message to the MESSAGE file during inventory management EXPROC processing. This message is issued for information only. The following information is in the message text:

volume_type This is the volume type and can be any of the following types:
PHYSICAL
LOGICAL
STACKED
RM_VRSEL
RM_EXPDT
TOTAL

number Number of *volume_type* volumes read during inventory management

percent Percentage of total volumes read by DFSMSrmm inventory management processing

System action DFSMSrmm inventory management processing continues

Operator response None

System programmer response
None

Source DFSMSrmm

Detecting Module EDGMUPD

6

Setting up an RMMplex

This chapter explains how to set up your systems so that you can use a single Data Facility System Managed Storage removable media manager (DFSMSrmm) control data set (CDS) on multiple systems without having a shared direct access storage device (DASD) environment. It provides detailed information about the implementation and use of an RMMplex.

This chapter includes the following information:

- Client/server terminology
- Implementing client and server systems
- Starting DFSMSrmm in an RMMplex
- Managing volumes in an RMMplex
- Operator commands

By the end of this chapter, you will understand the different vital record specification (VRS) types and how to create them. Sample VRS definitions are provided for a better understanding.

6.1 RMMplex

An RMMplex can optionally include one or more RMM subsystems as servers, and can include one or more client subsystems in addition to standard subsystems. The server subsystems and standard subsystems have direct access to and share the DFSMSrmm control data set. The client systems have no direct access to the DFSMSrmm control data set, but share the CDS via the server. All systems that share a DFSMSrmm control data set in this way are part of the same RMMplex.

We use the configuration that is shown in Figure 6-1, which shows a complex RMMplex implementation with two sysplexes and one non-plex sharing a single DFSMSrmm CDS. The second non-plex shows a normal shared DFSMSrmm CDS. Each plex has its own DASD environment that is not shared between the other plexes and each plex has a unique MVS catalog environment.

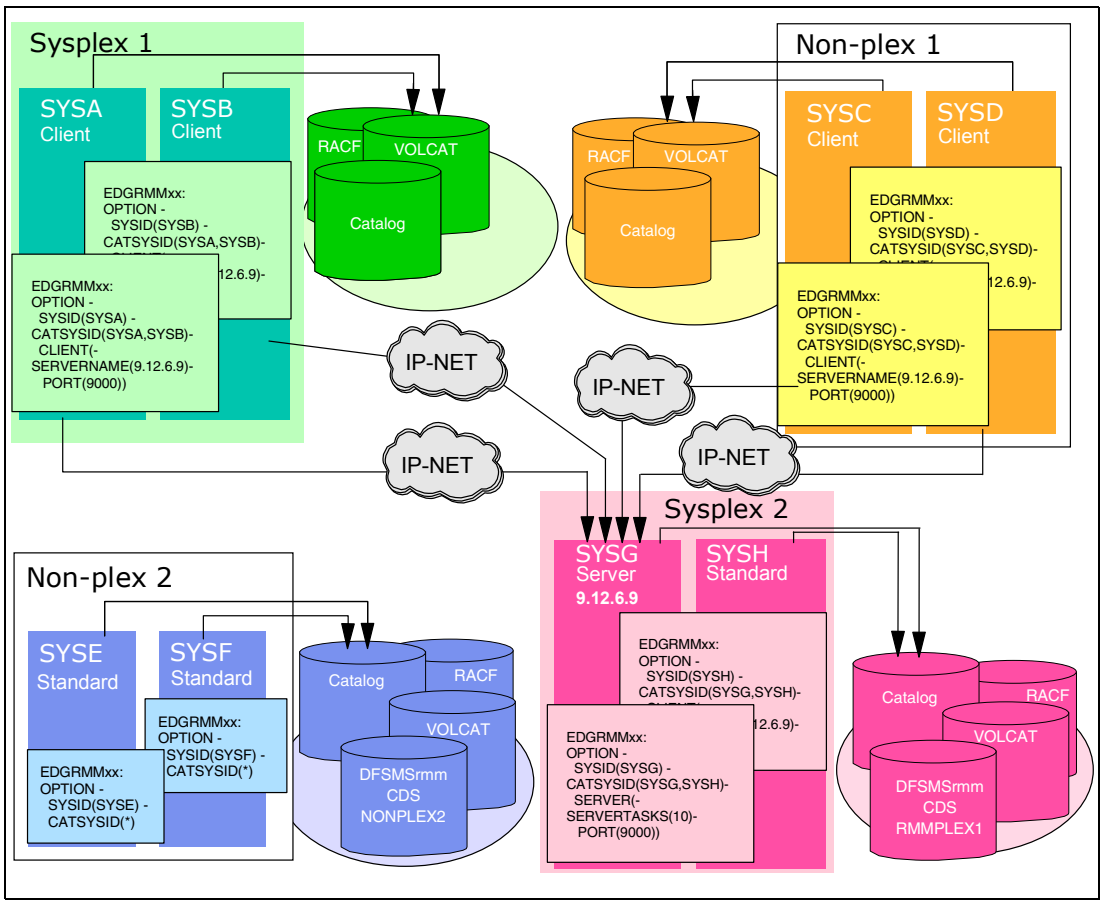


Figure 6-1 RMMplex with six systems sharing one DFSMSrmm CDS

Note: All dates and times that are used on the client system are local time. All dates and times that are used on the server are in local time. When data is retrieved from the RMM CDS and displayed by subcommand processing, the dates and times used are exactly as stored in the DFSMSrmm control data set. No conversion from server time zone to client time zone is performed.

6.1.1 Maintenance levels

The DFSMSrmm clients and servers can be on different releases. The minimum level is z/OS V1R6. As with CDS record toleration, DFSMSrmm provides toleration for future changes in the requests that are passed between systems. The record mappings for the requests remain restricted materials of IBM.

The server checks the version of the request data area and accepts it for processing if the version number matches that known to the server. If the level is higher on the client, the server fails the request with a new, specific return code and reason code combination. When a new level of data area is required, the level is set only into the data area when the new fields are used, so that an up-level client can perform non-new functions via a lower-level server.

Figure 6-2 shows an example of an Internet Protocol V6 (IPV6) address.

```
1080:0:0:0:8:800:200C:417A
```

Figure 6-2 IPV6 address

IPV6 supports networking strategy

The information in this section is important if you are planning to implement DFSMSrmm client and server systems or if you have implemented them.

DFSMSrmm on z/OS V1R12 and later releases is an IPv6-enabled application that supports both IPv4 and IPv6 sockets. You can continue to use IPv4 on all systems or you can run a mixed environment with one or more V1R12 systems using IPv6 and other systems using IPv4. After all systems are V1R12, you have the option of moving all systems to IPv6. In a mixed environment, dual-mode IP stacks are required. You can use IP addresses that are compliant with either IPV4 or IPV6.

Note: DFSMSrmm client/server processing is dependent on Internet Protocol V4.

6.2 Client/server terminology

We explain the terms client and server in a DFSMSrmm environment in the following sections.

6.2.1 DFSMSrmm server

A *server* in DFSMSrmm terminology is a system that runs when MVS and DFSMSrmm are up and active. This DFSMSrmm has direct access to the DFSMSrmm control data set and supports requests that come from other systems via TCP/IP.

6.2.2 SERVERNAME

The server name can be specified either by an IP address, a fully qualified domain name, or the host name of a server. DFSMSrmm uses the Domain Name System (DNS) to resolve a domain name or host name into an IP address. The values for the host name must conform to the following requirements:

- They must have a maximum of 63 characters.
- They must contain one or more tokens that are separated by a period.

- Each token must be larger than one character.
- The first character in each token must start with a letter.
- The remaining characters in each token must be a letter, number, or hyphen.

6.2.3 PORT

The port indicates the port number that is used for IP communication. You can specify a value of 1 - 65535. The port number must be the same for the client system and the server system to establish a network connection.

Note: Port numbers 1 - 1023 are reserved and must not be used.

6.2.4 SERVERTASKS operand

Use the SERVERTASKS operand to set the number of DFSMSrmm tasks that are available on the server to handle socket connections from client systems. You can specify a value of 1 - 999. DFSMSrmm uses this number to determine the number of tasks to be started for processing all client requests on this server.

Suggestion: Depending on the number of client systems, increase this value to allow three tasks per client. You do not need more server tasks than the sum of local tasks across your client systems.

6.2.5 DFSMSrmm client

A *client* in DFSMSrmm terminology is a system that runs when MVS and DFSMSrmm are up and active. This DFSMSrmm has no direct access to the DFSMSrmm control data set and sends all requests to the DFSMSrmm control data set via TCP/IP to the “server” specified in the EDGRMMxx parmlib member.

6.2.6 LOCALTASKS

Use the LOCALTASKS operand to set the number of tasks that are available on each system for processing locally initiated requests. On a client system, LOCALTASKS is also the maximum number of tasks that can make a socket connection to the server. You can specify a value 1 - 999.

6.3 Implementing client and server systems

Before you start, make sure that all systems are z/OS V1.6 or later. You must also have installed Internet Protocol V4.

Note: You can share the CDS with other systems that run any supported level of DFSMSrmm with any supported level of DFSMSrmm that has the appropriate toleration maintenance installed.

You can convert existing DFSMSrmm systems to be either client or server systems, or add new DFSMSrmm systems to an RMMplex. In addition, you can merge existing DFSMSrmm

systems into an RMMplex by merging the CDSs as explained in 15.6, “Merging DFSMSrmm environments” on page 614.

You can get the IP address of your DFSMSrmm server by using commands.

6.3.1 Checking your TCP/IP configuration

Before you can update the EDGRMMxx parmlib member to define your systems as DFSMSrmm client or server, obtain the necessary TCP/IP address of the server and test to ensure that the TCP/IP configuration is working.

HOMETEST command

When you are using the TSO HOMETEST command, run the IBM MVS TCP/IP CS V1R4 TCP/IP configuration tester to obtain the TCP/IP information. See Figure 6-3.

```
TSO HOMETEST
```

Figure 6-3 TSO HOMETEST command

Example 6-1 shows the result of the TCP/IP configuration tester.

Example 6-1 TCP/IP configuration tester output

```
Running IBM MVS TCP/IP CS V1R6 TCP/IP Configuration Tester

The FTP configuration parameter file used will be SYSFTPD DD.
TCP Host Name is: WTSC64

Using Name Server to Resolve WTSC64
The following IP addresses correspond to TCP Host Name: WTSC64
9.12.6.9

The following IP addresses are the HOME IP addresses defined in PROFILE.TCPIP:
9.12.6.9
9.12.6.31
10.1.101.64
10.1.101.64
10.1.101.64
127.0.0.1

All IP addresses for WTSC64 are in the HOME list!

Hometest was successful - all Tests Passed!
```

DISPLAY TCPIP command

You can also use either of the DISPLAY TCPIP MVS commands that are shown in Figure 6-4 to access the required TCP/IP information.

```
DISPLAY TCPIP,,NETSTAT,HOME
DISPLAY TCPIP,TCPIP,NETSTAT,HOME
```

Figure 6-4 DISPLAY TCPIP commands

A third way to obtain the required TCP/IP information is to use the TSO NETSTAT command, as shown in Figure 6-5 on page 196.

```
TSO NETSTAT,HOME
```

Figure 6-5 NETSTAT command

Example 6-2 shows the result of the DISPLAY TCPIP or TSO NETSTAT command.

Example 6-2 DISPLAY TCPIP or TSO NETSTAT command output

```
EZD0101I NETSTAT CS V1R6 TCPIP 587
HOME ADDRESS LIST:
LINKNAME:   OSA2CC0LNK
  ADDRESS:  9.12.6.9
    FLAGS:  PRIMARY
LINKNAME:   STAVIPA1LNK
  ADDRESS:  9.12.6.31
    FLAGS:
LINKNAME:   EZASAMEMVS
  ADDRESS:  10.1.101.64
    FLAGS:
LINKNAME:   IQDIOLNK0A016540
  ADDRESS:  10.1.101.64
    FLAGS:
LINKNAME:   EZAXCF63
  ADDRESS:  10.1.101.64
    FLAGS:
LINKNAME:   LOOPBACK
  ADDRESS:  127.0.0.1
    FLAGS:
INTFNAME:   LOOPBACK6
  ADDRESS:  ::1
    TYPE:   LOOPBACK
    FLAGS:
7 OF 7 RECORDS DISPLAYED
```

OMVS console

The third way to obtain TCP/IP information for your configuration is to use the Open MVS console commands. You can start the Open MVS console session with the TSO OMVS command, as shown in Figure 6-6.

```
TSO OMVS
```

Figure 6-6 OMVS command

After you enter the OMVS command, you see the information that is shown in Example 6-3. You can enter any Open MVS command. To access the TCP/IP configuration, use the HOSTNAME command.

Example 6-3 Open MVS command HOSTNAME

```
IBM
Licensed Material - Property of IBM
5694-A01 (C) Copyright IBM Corp. 1993, 2004
(C) Copyright Mortice Kern Systems, Inc., 1985, 1996.
(C) Copyright Software Development Group, University of Waterloo, 1989.

All Rights Reserved.
```

```
U.S. Government users - RESTRICTED RIGHTS - Use, Duplication, or
Disclosure restricted by GSA-ADP schedule contract with IBM Corp.

IBM is a registered trademark of the IBM Corp.

/usr/lpp/skrb/bin:/usr/lpp/dce/bin:/usr/lpp/ldap/bin:/usr/lpp/ldap/sbin:/usr/lpp
/java/J1.4/bin:/bin:/usr/sbin:.
MHLRES4 @ SC64:/>

 ===> HOSTNAME
                                                                        INPUT
ESC=¢   1=Help      2=SubCmd    3=HlpRetrn  4=Top       5=Bottom    6=TSO
        7=BackScr   8=Scroll    9=NextSess 10=Refresh  11=FwdRetr  12=Retrieve
```

Example 6-4 shows the result of the DISPLAY HOSTNAME command.

Example 6-4 Display HOSTNAME output

```
WTSC64OE.itso.ibm.com
MHLRES4 @ SC64:/>

 ===>
                                                                        INPUT
ESC=¢   1=Help      2=SubCmd    3=HlpRetrn  4=Top       5=Bottom    6=TSO
        7=BackScr   8=Scroll    9=NextSess 10=Refresh  11=FwdRetr  12=Retrieve
```

To test the TCP/IP connection between your current system and another system, enter the Open MVS command PING *hostname* command as shown in Example 6-5.

Example 6-5 Open MVS PING command

```
IBM
Licensed Material - Property of IBM
5694-A01 (C) Copyright IBM Corp. 1993, 2004
(C) Copyright Mortice Kern Systems, Inc., 1985, 1996.
(C) Copyright Software Development Group, University of Waterloo, 1989.

All Rights Reserved.

U.S. Government users - RESTRICTED RIGHTS - Use, Duplication, or
Disclosure restricted by GSA-ADP schedule contract with IBM Corp.

IBM is a registered trademark of the IBM Corp.

/usr/lpp/skrb/bin:/usr/lpp/dce/bin:/usr/lpp/ldap/bin:/usr/lpp/ldap/sbin:/usr/lpp
/java/J1.4/bin:/bin:/usr/sbin:.
MHLRES4 @ SC64:/>


 ===> PING WTSC63OE.itso.ibm.com
                                                                        INPUT
ESC=¢   1=Help      2=SubCmd    3=HlpRetrn  4=Top       5=Bottom    6=TSO
        7=BackScr   8=Scroll    9=NextSess 10=Refresh  11=FwdRetr  12=Retrieve
```

Example 6-6 shows the result of the PING command.

Example 6-6 Display PING output

```
MHLRES4 @ SC64:/>ping WTSC63OE.itso.ibm.com
CS V1R6: Pinging host WTSC63OE.itso.ibm.com (9.12.6.71)
```

```
Ping #1 response took 0.001 seconds.
MHLRES4 @ SC64:/>

 ===>
                                                                                INPUT
ESC=¢   1=Help      2=SubCmd    3=HlpRetrn  4=Top       5=Bottom    6=TSO
        7=BackScr   8=Scroll    9=NextSess 10=Refresh  11=FwdRetr  12=Retrieve
```

Note: Ensure that TCP/IP definition files are updated to identify the server host name, IP address, and port number.

To leave the Open MVS console environment, use the EXIT command.

Port reservation

Be sure that the port you are specifying in the DFSMSrmm EDGRMMxx parmlib member is not used by other tasks. Reserve a port in TCP/IP by updating the TCP/IP parameters as shown in Example 6-7.

Note: There is no need to reserve a PORT in TCP/IP. However, if you do not make a reservation, another task can use the same port number for its requests and your request to use the port number fails.

Example 6-7 Sample TCP/IP profile member

```
; PROFILE.TCPIP
; =============
PORT
    20 TCP OMVS                ; OE FTP Server
            DELAYACKS          ; Delay transmission acknowledgements
    21 TCP FTPMVS1 BIND 9.12.6.9  ; control port
    21 TCP FTPOE1  BIND 9.12.6.31 ; control port
    23 TCP INTCLIEN            ; MVS Telnet Server
    23 TCP OMVS BIND 9.12.6.31 ; OE Telnet Server
    80 TCP OMVS                ; OE Web Server
   111 TCP PMAP                ; Portmap Server
   111 UDP PMAP                ; Portmap Server
   135 UDP LLBD                ; NCS Location Broker
   161 UDP SNMPD               ; SNMP Agent
   162 UDP SNMPQE              ; SNMP Query Engine
   512 TCP REXECD BIND 9.12.6.9  ; Remote Execution Server
   512 TCP OMVS BIND 9.12.6.31 ; Remote Execution Server
   514 TCP REXECD BIND 9.12.6.9  ; Remote Execution Server
   514 TCP OMVS BIND 9.12.6.31 ; Remote Execution Server
   580 UDP NCPROUT             ; NCPROUTE Server
   750 TCP MVSKERB             ; Kerberos
   750 UDP MVSKERB             ; Kerberos
   751 TCP ADM@SRV             ; Kerberos Admin Server
   751 UDP ADM@SRV             ; Kerberos Admin Server
  2000 TCP IOASRV              ; OSA/SF Server
  2049 UDP NFS                 ; NFS Server
  3000 TCP CICSTCP             ; CICS Socket
  3333 TCP HWS                 ; IMS/OTMA
  3334 TCP HWSB                ; IMS/OTMA
  3335 TCP EZAIMSJL            ; IMS Listener BMP
  5001 TCP TPS                 ; OS/390 TME 10 GEM Connection Service
  9000 TCP DFRMM               ; DFSMSrmm RMMplex
```

```
INCLUDE TCP.COMMON.TCPPARMS(VIPAPORT)
```

NSLOOKUP command

You can use the NSLOOKUP command to specify an individual query in command mode or to issue multiple queries in interactive mode.

To issue a query, use the NSLOOKUP command in the format that is shown in Figure 6-7.

```
NSLOOKUP---.-domain_name----.---.----------------.-----!SubCommand!
           !-domain_address-!   !-server_name----!
                                !-server_address-!
```

Figure 6-7 NSLOOKUP command syntax

The parameters and subcommands of NSLOOKUP are case-sensitive and must be entered in lowercase. Parameter values and domain names are not case sensitive.

Use the NSLOOKUP command that is shown in Figure 6-8 to determine the full server name if you specify the server IP address. Alternatively, you can specify the server name to get the server address.

```
TSO NSLOOKUP 9.149.157.65
```

Figure 6-8 NSLOOKUP command

Example 6-8 shows the result of the NSLOOKUP command.

Example 6-8 NSLOOKUP command result

```
Server:  ips-de-a.de.ibm.com
Address:  9.165.1.10
Name:    stutvs.megacenter.de.ibm.com
Address:  9.149.157.65
```

Command mode query

A *command mode query* is entered by supplying a complete query, including any desired options and the desired name server, if different from the default name server, with the NSLOOKUP command.

Operands

For a command-mode query, use the following parameters:

DOMAIN_NAME Queries the name server for information about the current query type of domain_name. The default query type is A (address query).

DOMAIN_ADDRESS Reverses the components of the address and generates a pointer type (PTR) query to the name server for the "in-addr.arpa" domain mapping of the address to a domain name.

SERVER_NAME Directs the default name server to map server_name to an IP address and then use the name server at that IP address.

SERVER_ADDRESS Specifies the IP address of the name server to be queried other than the default name server. A query for the address in the "in-addr.arpa" domain is initially made to the default name server to map the IP address to a domain name for the server.

6.3.2 EDGRMMxx DFSMSrmm options

Update your existing EDGRMMxx parmlib member with the CLIENT and SERVER operands and select appropriate values for the suboperands. Also, check the CATSYSID and CDSID operands. If these operands are not already set, add them now.

This section describes only the operands that are related to the DFSMSrmm client/server support.

CATSYSID

Specify CATSYSID to enable DFSMSrmm catalog synchronization. Use the CATSYSID operand to identify the user catalogs you want tracked when you run the DFSMSrmm EDGHSKP utility with the CATSYNCH operand to use catalog status tracking. All the systems that you identify must have the z/OS version of that code that supports catalog synchronization. See Chapter 9, “Catalog synchronization” on page 329, for more information. The operands are explained:

CATSYSID(*)
This means that all catalogs are fully shared. You must specify an asterisk (*) to specify that catalogs are fully shared so that any data set can be processed by DFSMSrmm on any DFSMSrmm subsystem.

CATSYSID(*sys_ID_list*)
This provides a list of DFSMSrmm system IDs that share tape data set user catalogs with this DFSMSrmm subsystem. You can identify up to 16 system IDs for DFSMSrmm subsystems. You must specify a list of system IDs when user catalogs are not shared. Ensure that they are synchronized with the DFSMSrmm CDS before you run inventory management vital record processing. Include IDs for systems that you no longer use but for which you are still retaining tape data sets.

The default is None.

CDSID

The CDSID option specifies the identifier of the CDS that must be used on this system. Specify a value one to eight alphanumeric characters long. When you start DFSMSrmm, the CDSID value is compared to the ID in the CDS control record.

If the IDs match, DFSMSrmm start-up continues. If the CDS does not have an ID, DFSMSrmm creates the ID in the control record from the CDSID. If the IDs do not match, DFSMSrmm start-up fails and DFSMSrmm issues a message to the operator to select another parmlib member. If you do not specify a value for CDSID, you cannot use a CDS that already has an ID in its control record.

See the “Creating or Updating the Control Data Set Control Record” chapter in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405, for information about how the DFSMSrmm EDGUTIL utility sets the CDS ID.

The default is None.

CLIENT

The CLIENT option specifies the type of system you want to set up. CLIENT is mutually exclusive with SERVER. If neither client nor server is specified, DFSMSrmm starts as a standard system.

The operands are explained:

SERVERNAME(*servername*)

SERVERNAME is a required operand when you specify CLIENT. The *servername* can either be an IP address, a fully qualified domain name, or a server host name.

An IP address can be specified either as a hexadecimal IPv6 address or a numeric IPv4 address. Specify an IPv6 IP address only when the server is running z/OS V1R12 or a later release and TCP/IP on the server system has IPv6 dual-mode IP stack. For example, CLIENT(SERVERNAME(1080:0:0:0:8:800:200C:417A)) specifies an IPv6 IP address. For the syntax of an IPv6 IP address, see RFC3513 at this website:

`http://tools.ietf.org/html/rfc3513`

When a domain name or a host name is specified, DFSMSrmm uses the Domain Name System (DNS) to resolve that name into an IP address. The host name can be a maximum of 63 characters.

The host name must contain one or more tokens that are separated by a period. Each token must be larger than one character. The first character in each token must start with a letter. The remaining characters in each token can be a letter, number, or hyphen.

`CLIENT(SERVERNAME(RMMPLX1.MAINZ.IBM.COM) PORT(1950))`, for example, tells DFSMSrmm to start as a client without direct DASD access and to share the tape inventory being accessed by the RMM server with the host name RMMPLX1 using network IP protocol port 1950.

PORT(*PortNumber*)

Use this operand to specify the port number to be used for IP communication. The PORT operand is required. Specify a value of 1 - 65535. Port numbers 1 - 1023 are reserved. Also, the client port number and server port number must match for the systems to communicate.

The default is None.

SERVER

This option specifies the type of system you want to set up. SERVER is mutually exclusive with CLIENT. Neither SERVER nor CLIENT must be specified when DFSMSrmm is used as a standard system.

The operands are explained:

PORT(*PortNumber*)

Use this operand to specify the port number to be used for IP communication. The PORT operand is required. Specify a value of 1 - 65535. Port numbers 1 - 1023 are reserved. The port number must be the same for the client system and the server system to establish a network connection.

The default is None.

SERVERTASKS(*number*)

Use this operand to specify how many DFSMSrmm tasks must be available on the server to handle socket connections from client systems. DFSMSrmm uses this number to determine how many tasks

are to be started for processing all client requests on this server. Specify a value of 1 - 999.

The number of local and server tasks you can use and still successfully start DFSMSrmm is limited by the size of the private region above and below 16 MB. To start with more tasks, you need a larger REGION size.

The default is 10.

Suggestion: Specify or accept the default value of 10 server tasks on each server system. Depending on the number of CLIENT systems, increase this value to allow three tasks per CLIENT. Most of the local tasks are rarely used by DFSMSrmm. You do not need more server tasks than the sum of local tasks across your CLIENT systems.

6.3.3 Updating EDGRMMxx for a client

In the environment shown in Figure 6-1 on page 192, we define four DFSMSrmm client systems: two in SYSPLEX 1 and two in Non-Plex 1. To implement a DFSMSrmm client system, update your EDGRMMxx parmlib member to define the CATSYSID, CDSID, and CLIENT operands. You can use the same DFSMSrmm client definitions as shown in Figure 6-9 for SYSA and SYSB. All other EDGRMMxx parmlib definitions are not shown in Figure 6-9.

```
OPTION OPMODE(P)                          /* operating mode      */ -
       CATSYSID(SYSA,SYSB)                /* Catalog sysid(s)    */ -
       CDSID(RMMPLEX1)                    /* CDS ID              */ -
       CLIENT(SERVERNAME(9.12.6.9) PORT(9000)) /* WTSC Server    */ -
       LOCALTASKS(10)                     /* number of subtasks  */
```

Figure 6-9 EDGRMMxx parmlib options for clients SYSA and SYSB

For both systems SYSC and SYSD, you can use the CATSYSID, CDSID, and CLIENT operands as shown in Figure 6-10. All other EDGRMMxx parmlib definitions are not shown in Figure 6-10.

```
OPTION OPMODE(P)                          /* operating mode      */ -
       CATSYSID(SYSC,SYSD)                /* Catalog sysid(s)    */ -
       CDSID(RMMPLEX1)                    /* CDS ID              */ -
       CLIENT(SERVERNAME(9.12.6.9) PORT(9000)) /* WTSC Server    */ -
       LOCALTASKS(10)                     /* number of subtasks  */
```

Figure 6-10 EDGRMMxx parmlib options for clients SYSC and SYSD

6.3.4 Updating EDGRMMxx for a server

As shown in Figure 6-1 on page 192, we have only one DFSMSrmm server in SYSPLEX 2 on system SYSG. To implement a DFSMSrmm server system, update your EDGRMMxx parmlib member as shown in Figure 6-11 on page 203 to define the CATSYSID, CDSID, and SERVER operands. All other EDGRMMxx parmlib definitions are not shown in Figure 6-11 on page 203.

```
OPTION OPMODE(P)                          /* operating mode      */ -
       CATSYSID(SYSG,SYSH)                /* Catalog sysid(s)    */ -
       CDSID(RMMPLEX1)                    /* CDS ID              */ -
       SERVER(PORT(9000) SERVERTASKS(12)) /* for 4 clients       */ -
       LOCALTASKS(10)                     /* number of subtasks  */
```

Figure 6-11 EDGRMMxx parmlib options for server SYSG

6.4 Updating the DFSMSrmm CDSID (optional)

Update the CDS control record to specify a unique CDS identifier (CDSID) that must be specified in each EDGRMMxx parmlib member using this DFSMSrmm CDS.

6.4.1 Sample EDGUTIL job control

To set or update a CDSID in the DFSMSrmm CDS control record, use the EDGUTIL utility with a parameter of UPDATE.

Note: To create or update the control record, the user of EDGUTIL must have UPDATE or higher RACF access to the DFSMSrmm CDS.

Example 6-9 shows a sample JCL that you can use to set or update the DFSMSrmm CDS control record with a CDSID of RMMPLEX1.

Example 6-9 Sample EDGUTIL job control

```
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
 CONTROL CDSID(RMMPLEX1)
/*
```

CDSID

This option specifies one-to-eight alphanumeric characters that identify the CDS by name. There is no default.

At DFSMSrmm start-up time, DFSMSrmm matches this CDSID value with the CDSID operand in parmlib member EDGRMMxx. The CDSID value in EDGUTIL sets or changes the CDS ID. EDGUTIL does not validate the CDSID in the CDS control record. It simply sets the new value into the control record.

If you do not specify an ID for the CDS, you can set it the first time DFSMSrmm is started with this CDS by setting the CDSID operand in EDGRMMxx.

Do not change the CDSID value after you set it, because changing it can affect inventory management. You can change the value only when no data sets exist that were created on the system or when the WHILECATALOG option for VRSs is not in use.

6.5 Updating your firewall (optional)

You must update your firewall to ensure that communication between DFSMSrmm clients and servers is allowed only for the defined IP addresses and ports. The DFSMSrmm subsystem does no authentication, encryption, or verification of connect requests that are received on the server other than to verify that it is a valid DFSMSrmm request and that CDS IDs match. Also, consider using RACF to protect the use of the IP addresses defined for DFSMSrmm and limit the use of the IP address to the DFSMSrmm started task.

6.6 Starting DFSMSrmm in an RMMplex

After you update the EDGRMMxx parmlib member and you optionally check your firewall definitions, start or restart DFSMSrmm on your server and client.

Note: By default, the RMM subsystem address space starts unchanged, as a standard system, unless one of the new parmlib command operands is used to identify the system as either a client or a server.

6.6.1 Starting and restarting DFSMSrmm on your client and server

To start DFSMSrmm with the default procedure name and parmlib member, enter the MVS command that is shown in Figure 6-12.

```
S DFRMM
```

Figure 6-12 DFSMSrmm start command

If you want, start DFSMSrmm with different parameters than the default. Figure 6-13 shows how you can use the MVS command.

```
S DFRMM,M=xx,DSN=parmlib_dataset
```

Figure 6-13 DFSMSrmm start command with options

The following explanations apply to Figure 6-13:

M Specify the suffix of the EDGRMMxx member in parmlib
DSN Specify a different parmlib data set name

After this command, you see the message that is shown in Figure 6-14.

```
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 6-14 EDG0105I initialization message

If the subsystem interface was not activated before the command, you see the message that is shown in Figure 6-15 on page 205.

```
EDG0103D DFSMSrmm SUBSYSTEM INTERFACE IS INACTIVE - ENTER "IGNORE",
         "CANCEL" OR "RETRY"
```

Figure 6-15 EDG0103D initialization message

Enter `RETRY` to continue with the initialization. After that, you must receive the EDG0105I message.

You can restart DFSMSrmm with the MVS MODIFY command. For example, to restart DFSMSrmm and implement new parmlib options, specify the command that is shown in Figure 6-16.

```
F DFRMM,M=01
```

Figure 6-16 Modify DFRMM command

The following explanation applies to Figure 6-16:

M Specifies the suffix of the EDGRMMxx member in parmlib

After you enter this command, you see the initialization completion message as shown in Figure 6-17.

```
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 6-17 EDG0105I initialization message after restart

Consideration for the DFSMSrmm server start

When a server subsystem starts, identify some basic information for TCP/IP to the DFSMSrmm subsystem with the EDGRMMxx parmlib OPTION SERVER operand. After the start-up of the standard subsystem, DFSMSrmm communicates with TCP/IP and prepares to handle DFSMSrmm requests from client systems.

The server verifies that TCP/IP and the specified PORT are available, determines its own default IP address, and informs the operator that initialization is successful or issues error messages. In addition to normal DFSMSrmm subsystem operation, such as processing local requests, the server waits for and accepts connection requests. It also processes requests from DFSMSrmm client systems. If the server task is unable to start successfully, you can still use DFSMSrmm as a standard subsystem until the server task problem is resolved.

Consideration for the DFSMSrmm client start

When a client subsystem starts, identify some basic information for TCP/IP to the DFSMSrmm subsystem via the EDGRMMxx parmlib OPTION client operand.

During start-up, the subsystem communicates with TCP/IP and prepares to send DFSMSrmm requests from the client to a server system. The client verifies that TCP/IP is available and that the server can be reached with the specified PORT, determines its own IP address, verifies the CDS ID matches that of the server, and informs the operator that initialization is successful or issues error messages.

DFSMSrmm ignores any parmlib options that are not required for a client system. The client can connect to only one server system at a time. If the defined server is not available, the client can issue WTOR EDG0358D and wait for either the operator to reply with CANCEL, RETRY, or for the server to be available for connection.

If the server is not available, DFSMSrmm fails all in-progress requests with an I/O error that requires server processing when a refresh (F DFRMM,M=xx) is requested. Existing error recovery in DFSMSrmm can handle this and provide recovery when any new server is available.

DFSMSrmm issues a WTOR when a TCP/IP error occurs. RETRY processing relies on the operator replying to the WTOR. If the error is resolved and the operator has not replied to the WTOR, DFSMSrmm processing automatically continues and cancels the outstanding WTOR. When the client is started, no further verification of the server availability is performed unless a DFSMSrmm request is to be processed.

It might happen that a request is processed and server communication fails or a timeout occurs, and retry still cannot process the request. In this case, a WTOR, DFSMSrmm issues the message EDG0358D to describe the error. It prompts the operator to reply CANCEL or RETRY, and automatic retry by DFSMSrmm is possible.

When the CLIENT is started, no further verification of the SERVER availability is performed unless an RMM request is to be processed (no heartbeat is maintained). When a request is processed and SERVER communication fails or a timeout occurs, and retry still cannot process the request, a WTOR EDG0358D message is issued that details the error and allows operator involvement in error recovery. CANCEL or RETRY is possible.

6.7 Managing volumes in an RMMplex

You can have different environments in an RMMplex. For example, you can have fully shared catalogs, unshared catalogs, a shared RACF database, or an unshared RACF database. The following section describes the differences.

6.7.1 Catalogs in an RMMplex

You can implement DFSMSrmm client/server support regardless of whether your catalogs are fully shared across all of the systems in an RMMplex. However, you must correctly identify to DFSMSrmm whether catalogs are shared. It is your choice whether you synchronize the catalogs with the DFSMSrmm CDS. The catalog status for all data sets is maintained in the server system CDS.

Based on unshared catalogs between client and server, the catalogs and the CDS must be synchronized if catalog retention or the UNCATALOG(Y/S) option is used. To synchronize the CDS and the catalogs, use CATSYSID(sys_ID_list) and follow the process for unshared catalogs. You can run EDGHSKP with CATSYNCH, VERIFY, and EXPROC on the client system.

With unshared catalogs, the UNCATALOG(YES) parmlib option, on the server system, cannot be honored for data sets that are created on the client systems. The server has no way to communicate with the client to initiate uncatalog processing. However, when processing is requested from the client, there is special recognition and handling of the request so that any catalog or RACF profile updates can be initiated on the client system.

DFSMSrmm uncatalogs data sets on a volume when you specify the EDGRMMxx PARMLIB OPTION UNCATALOG(Y) command under these conditions:

- A volume is returned to scratch status, and DFSMSrmm uncatalogs all the data sets on the volume.
- The RMM DELETEVOLUME FORCE subcommand is issued for a volume, and the DFSMSrmm uncatalogs all the data sets on the volume.
- The RMM CHANGEVOLUME DSNAME subcommand is issued for a volume, and DFSMSrmm uncatalogs all the data sets on the volume. If the data set name specified on the RMM CHANGEVOLUME subcommand matches the data set name on the volume, DFSMSrmm only uncatalogs subsequent data sets.
- The RMM DELETEDATASET subcommand is issued for a data set, and DFSMSrmm uncatalogs the data set. Also, DFSMSrmm uncatalogs all data sets recorded on the same volume with higher data set sequence numbers.
- A tape data set is overwritten, and DFSMSrmm uncatalogs the data set. Also, all data sets recorded on the same volume with higher data set sequence numbers are uncataloged.

DFSMSrmm uncatalogs data sets on a volume when you specify the EDGRMMxx PARMLIB OPTION UNCATALOG(S) command. It does this only under the condition that, when a volume is returned to scratch status, DFSMSrmm uncatalogs all the data sets on the volume.

The default is UNCATALOG(N) unless DFSMSrmm is running in protect mode. DFSMSrmm uses the UNCATALOG default of Y when DFSMSrmm is running in protect mode.

For information about how to synchronize your MVS catalogs, see Chapter 9, “Catalog synchronization” on page 329.

6.7.2 RACF considerations in an RMMplex

You might have switched on RACF TAPEVOL or TAPEVOL and TAPEDSN on your client and server. You might also have specified in the EDGRMMxx PARMLIB OPTION TPRACF(P) or TPRACF(A) to request DFSMSrmm tape security support. In this case, you can implement DFSMSrmm client/server support regardless of whether your RACF database is fully shared across all of the systems in an RMMplex. However, you must correctly identify to DFSMSrmm whether RACF database is shared.

Note: Check whether you can use the new SAF tape security. For more information about how to implement it, see Chapter 13, “System Authorization Facility tape security” on page 475.

With unshared RACF databases, the RACF TAPEVOL profiles and TVTOC entries are maintained only on the server system under these conditions:

- Confirm that you have returned the volume to its owner and the volume was protected by a RACF TAPEVOL profile.
- Confirm that you have erased the volume.
- Confirm that you have initialized the volume.
- Confirm that you have performed all activities necessary before returning the volume to scratch status.
- A volume is returned to scratch status, and DFSMSrmm deletes the RACF TAPEVOL profile.

- The RMM DELETEVOLUME FORCE subcommand is issued for a volume, and DFSMSrmm deletes the RACF TAPEVOL profile.
- The RMM CHANGEVOLUME DSNAME subcommand is issued for a volume, and DFSMSrmm deletes all the TVTOC entries for this volume. If the data set name that is specified on the RMM CHANGEVOLUME subcommand matches the data set name on the volume, DFSMSrmm only deletes TVTOC entries for subsequent data sets.
- The RMM DELETEDATASET subcommand is issued for a data set, and DFSMSrmm deletes the TVTOC entry. DFSMSrmm also deletes all TVTOC entries that are defined for the volumes with higher data set sequence numbers.
- A tape data set is overwritten and DFSMSrmm deletes the TVTOC entries. Also, all TVTOC entries on the same volume with higher data set sequence numbers are deleted.

For detailed RACF information, see Appendix A, “Security topics” on page 633.

6.7.3 System-managed libraries in an RMMplex

You can perform the following tasks for a system-managed tape library that is known to the client and not the server:

- Ejecting a volume from a system-managed tape library
- Adding volumes to a system-managed tape library with STATUS(VOLCAT)
- Changing volume attributes that are also maintained in the tape configuration database (TCDB)
- Running expiration processing
- Confirm moves for exported stacked volumes
- Running EDGUTIL VERIFY with the TCDB and optionally the Library Manager

6.7.4 HOUSEKEEP processing in an RMMplex

The utilities provide restricted functions when run on a client or server system. Before running EDGHSKP, you must define several data sets. The data sets that are used by both the EDGHSKP utility and the DFSMSrmm subsystem address space must be pre-allocated and cataloged.

EDGHSKP

You can run all inventory management functions except BACKUP on a client system. However, because there is no direct connection to the CDS, the elapsed times will be longer when run on a client system. We suggest that you run all inventory management functions other than CATSYNCH and EXPROC on either the server or a standard DFSMSrmm system.

Note: When you attempt backup on a client system, the utility ends with RC16 and DFSMSrmm issues message EDG6137E stating that it cannot be run on a client system.

EDGUTIL

When you run EDGUTIL on a client system, the utility can only process a DFSMSrmm CDS that is not in use by DFSMSrmm. Because there is no CDS for a client system, you cannot run with the active CDS. However, you can recover a backup copy of the DFSMSrmm CDS to the client system to run VERIFY(VOLCAT), VERIFY(SMSTAPE), and MEND(SMSTAPE).

When you run EDGUTIL on a server system, you cannot access the TCDB or library for tape libraries that you can access only from the client system. When you run EDGUTIL on a client system and do not specify the MASTER DD, DFSMSrmm issues message EDG6101E and the utility ends with return code 12.

EDGBKUP

You cannot process an active DFSMSrmm CDS on a client system. All functions are available on a client system if you provide the DD statements for the CDS and journal. This enables you to use the BACKUP and RESTORE options independently of the DFSMSrmm subsystem. When you run EDGBKUP on a client system and do not specify the MASTER DD or JOURNAL DD, DFSMSrmm issues message EDG6101E and the utility ends with return code 12.

Figure 6-18 shows the correct sequence in which to run your daily housekeeping jobs so that all components are updated correctly and are always in sync.

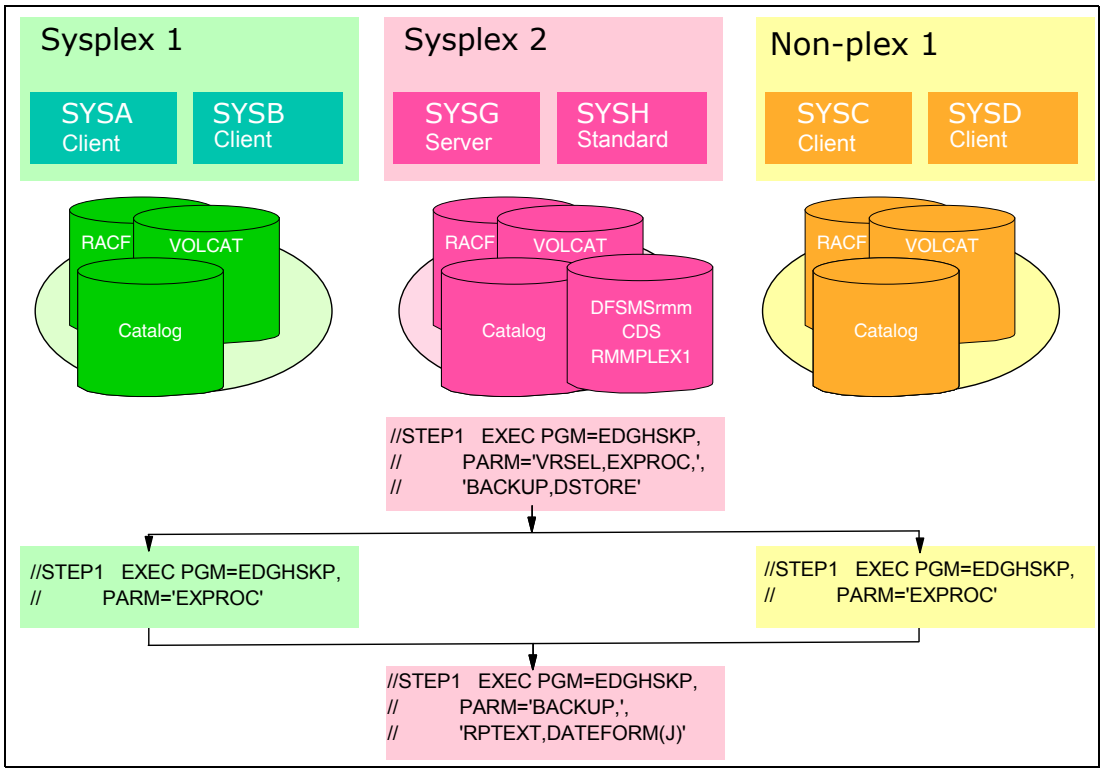


Figure 6-18 DFSMSrmm housekeeping in an RMMplex

6.8 Operator commands

The DFSMSrmm client system processes most requests by communicating with the server but also by processing those local requests that can be completely processed on the client system.

When multiple tasks are being processed, DFSMSrmm maintains a queue in first-in, first-out (FIFO) order. You can use the operator command MODIFY DFRMM,Q A to display the tasks and a summary of the queues. The DFSMSrmm server processes local requests from DFSMSrmm code on the server unchanged. In addition, client requests are accepted and processed synchronously while the requester on the client waits. There is no queue of client requests that are maintained on the server. The request queues maintained on the server are for local requests only. When you list the active tasks, using the QUERY ACTIVE command, the active local requests are listed together with the accepted client requests.

The QUERY ACTIVE command that is shown in Figure 6-19 displays the active and queued tasks. To also support IP server tasks, the Q A command lists those subtasks, as well, and includes them in the count of total subtasks, count of active subtasks, and a list of active subtasks with a summary of the requests they process: sysID, jobname, function code, status, and subtask token.

```
MODIFY DFRMM,QUERY ACTIVE
or
F DFRMM,Q A
```

Figure 6-19 DFRMM query active command

In addition, the output from Q A is changed to be a single, multiline WTO issued as a command response using CART/CONSID. Example 6-10 shows the result of the QUERY ACTIVE command.

Example 6-10 DFSMSrmm status display

```
EDG1119I DFSMSrmm STATUS IS ACTIVE. JOURNAL ENABLED.
EDG1114I LOCAL TASKS  10, ACTIVE   0, SERVER TASKS   0, ACTIVE   0
EDG1118I      0 QUEUED REQUESTS, INCLUDING      0 NOWAIT      0 CATALOG
EDG1121I DEBUG IS DISABLED, PDA TRACE LEVEL: 1-2-3-4-
EDG1101I DFSMSrmm MODIFY COMMAND ACCEPTED
```

In Example 6-10, the following variables are used:

sysID SYSID value from the client, or the local SYSID (local requests)

Jobname The job name of original requester; for client requests, the job name on the client system

Function code The RMM subsystem request function code, from the subsystem options block (SSOB). There is a new subsystem function code that is used specifically to handle client requests. All others are unchanged.

Status IP communication status. The status code can be Co (Connect), Wr (Write), Re (Read), or Cl (Close) (IUO: The status is taken from the QMCB.)

7

Tailoring your DFSMSrmm environment

This chapter describes how to implement additional function, which is provided in Data Facility System Managed Storage removable media manager (DFSMSrmm) or supports DFSMSrmm, and how to tailor DFSMSrmm to your environment.

This chapter contains the following information:

- Implement additional function that supports DFSMSrmm
- IBM Tivoli Workload Scheduler
- Setting up DFSMSrmm common time support
- Dynamic installation exits
- Maintaining your control data set
- Initializing and erasing volumes
- Disposition processing
- Software products and foreign tapes
- DFSMSrmm volume pools
- Deleting non-existent volumes
- Repairing actions

By the end of this chapter, you will have an understanding of the additional function in DFSMSrmm and how to customize it for your environment.

7.1 IBM Tivoli Workload Scheduler

To manage the resources defined to DFSMSrmm, you need to run the inventory management tasks, also known as *housekeeping*, periodically. DFSMSrmm provides the EDGHSKP utility to help you perform inventory management. A job scheduling product, such as IBM Tivoli Workload Scheduler for z/OS (which incorporates Tivoli Operations Planning and Control (OPC)), is helpful to run these tasks.

IBM Tivoli Workload Scheduler for z/OS is a system management product that functions as a batch job scheduler. You can configure the typical DFSMSrmm batch job flow to IBM Tivoli Workload Scheduler for z/OS to run inventory management jobs regularly. This provides the following benefits:

- Prevents more than one EDGHSKP job from running at the same time
- Avoids jobs failing because inventory management is already running
- Prevents EDGUTIL VERIFY jobs from running at the same time as an inventory management jobs
- Handles the dependencies between jobs

The sample IBM Tivoli Workload Scheduler for z/OS stream included in the DFSMSrmm SAMPLIB library proposes the following steps when running inventory management. The name in parentheses is the sample member name included in SAMPLIB:

- Preparation (EDGJHKPA)
- Control data set (CDS) verify (EDGJVFY)
- CDS backup (EDGJBKP1, EDGJBKP2)
- Inventory management (EDGJDHKP(daily), EDGJWHKP(weekly))
- Erasing and labeling volumes (EDGJINER)
- Scratch processing (EDGJEXP)
- Scratch reporting (EDGJSCR)
- Ejecting volumes from a system-managed tape library (EDGJEJC)
- Producing reports (EDGJMOVE)
- Move confirmation (EDGJCMOV)

Also included in SAMPLIB is the IBM Tivoli Workload Scheduler for z/OS batch loader sample JCL, EDGJLOPC.

7.1.1 IBM Tivoli Workload Scheduler overview

This section introduces the basic IBM Tivoli Workload Scheduler for z/OS concepts, which you need to be familiar with for a better understanding of the samples provided by DFSMSrmm. Note that the Tivoli OPC product has been renamed to *IBM Tivoli Workload Scheduler for z/OS*.

IBM Tivoli Workload Scheduler for z/OS includes the following components:

- **Calendars** contain the information about the working days and free days for work planning purposes.
- **Workstations** are the logical place where work occurs. There are three types of workstations:
  - Computer workstations
  - Printer workstations
  - General workstations

The majority of the IBM Tivoli Workload Scheduler for z/OS operations run in computer workstations. The activity that occurs at each workstation is known as an *operation*. Each operation has a *job name*, *workstation ID*, and *operation number*.

- **Applications** are units of production work that IBM Tivoli Workload Scheduler for z/OS schedules. Each application can be one or more operations.
- **Special resources** (optional) represent any type of limited resources, such as data sets, tape drives, or communication lines.

IBM Tivoli Workload Scheduler for z/OS can schedule jobs on a planned basis or dynamically using event-triggered scheduling. The *Event-Triggered Tracking* (ETT) gives you a method of controlling and tracking workload that cannot be planned in advance. Some of the situations that we can consider in this special scheduling are the journal full condition and low-on-scratch situation.

The DFSMSrmm batch loader sample helps you in the definition of applications and special resources. Calendars and workstations must be defined using the IBM Tivoli Workload Scheduler for z/OS dialog.

7.1.2 Scheduling your inventory management tasks

The schedule of inventory management tasks must be defined before running. You can schedule each DFSMSrmm utility to run on its own, or schedule many activities to run together in a sequence. Some activities need to be performed daily, such as vital records processing, while other activities can be performed less frequently.

Table 7-1 shows our suggestions, which can help you to decide how frequently to run DFSMSrmm utilities for your requirements.

Table 7-1 Schedule inventory management tasks

Activity	Frequency
Processing vital records daily	Daily
Performing expiration processing	Daily
Performing storage location management	Weekly
Creating an extract data set	Daily
Initializing and erasing volumes	Weekly
Creating security and audit reports	Monthly
Backing up the DFSMSrmm control data set (CDS)	Daily
Backing up the DFSMSrmm journal	Daily
Verifying the CDS	Monthly

Note: If you are using DFSMSrmm z/OS 1.13 or higher, you do not need to run VRSEL processing unless any volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle the expiration of all volumes managed by the EXPDT retention method.

We can use the IBM Tivoli Workload Scheduler for z/OS batch loader sample JCL, EDGJLOPC, to define the schedule for the activities shown in Table 7-1 on page 213. The sample JCL defines the following applications based on the execution frequency:

- Daily applications:
  - RMMBKP
  - RMMHKPD
  - RMMPOST
  - RMMEXP
- Weekly applications:
  - RMMWEEK
  - RMMMOVE
- Monthly application

  RMMMTH
- Other applications
- RMMVRSVER is run only when a previous application finished with RC=8.

Figure 7-1 shows the flow of the IBM Tivoli Workload Scheduler for z/OS batch loader sample.

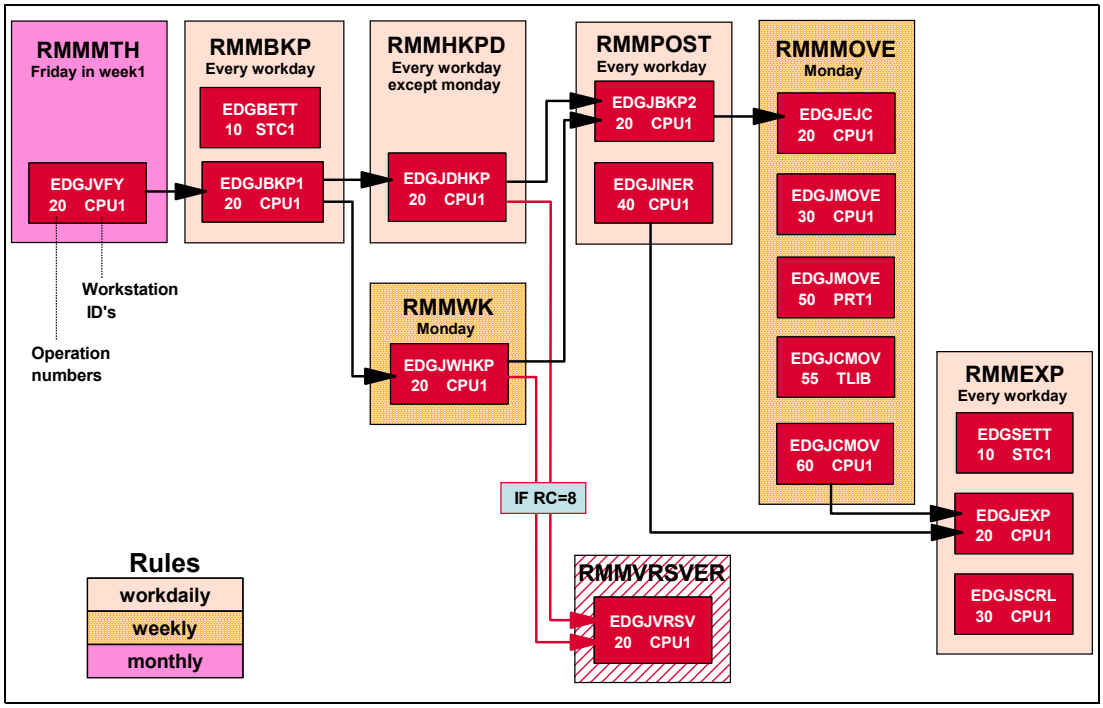


Figure 7-1 DFSMSrmm IBM Tivoli Workload Scheduler for z/OS job flow

In each application, you can see the different operations included. These are the jobs that perform each of the steps introduced at the beginning of the chapter. There are two more jobs that are defined under ETT: EDGBETT and EDGSETT. EDGBETT triggers the RMMEXP application if a low-on-scratch condition is reached, and EDGSETT triggers the RMMBKP application if the journal full threshold is reached.

When you have decided on your schedule, customize the EDGJLOPC sample to update the sample schedule to meet your needs. For more details, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

The following sections cover the different activities that are performed in each of the suggested steps that comprise the whole inventory management process.

7.1.3 Preparation

This first step is performed by sample job EDGJHKPA. This job executes the EDGPHKPA procedure for allocating the necessary inventory management files. Inside the EDGPHKPA procedure, there are some other procedures, each allocating a different file:

- EDGPMSGA: Allocates the (+1) version of the MESSAGE file
- EDGPVRSA: Allocates the (+1) versions of the REPORT and ACTIVITY files
- EDGPACTA: Allocates the files needed for the ACTIVITY reports:
  - REPORT.RUNINFO
  - REPORT.VRS
  - REPORT.VRS.SUMMARY
  - REPORT.RETDATE
  - REPORT.RETDATE.SUMMARY
  - REPORT.MATCHVRS
  - REPORT.MATCHVRS.SUMMARY
  - REPORT.SUBCHAIN
  - REPORT.SUBCHAIN.SUMMARY
- EDGPRPTA: Allocates the (+1) version of the REPTEXT file

This procedure also allocates the SCRDATE file.

7.1.4 CDS verify

A sample IBM Tivoli Workload Scheduler for z/OS job EDGJVFY is provided for verifying the CDS contents. The job invokes an EDGPVFY procedure, which uses the EDGUTIL program with PARM=VERIFY. The expected completion is the return code 0. You must perform manual recovery before dependent jobs can be run when the return code is greater than 0.

For more information about verifying the CDS, see 7.10.1, “Using the VERIFY parameter” on page 259.

7.1.5 CDS backup

Sample jobs EDGJBKP1 and EDGJBKP2 are provided in SYS1.SAMPLIB. These jobs are used to back up the CDS and journal, and to clear the journal after a successful backup. The jobs invoke the EDGPBKUP procedure, which executes the EDGHSKP program with PARM='BACKUP(*type*)', with *type* being one of DSS or AMS. If you select DSS, you can use concurrent copy for backing up the CDS. It also allocates the files that are needed for the backups.

EDGPMSGC, which allocates other files needed for the backup process, is another procedure inside the jobs.

7.1.6 Inventory management

DFSMSrmm manages the removable media resources and performs the inventory management (housekeeping) according to the policies defined to DFSMSrmm as vital record specifications (VRSs).

The inventory management processing includes the following tasks:

- **Inventory management processing (VRSEL)**

  The purpose of vital records processing is to identify the volumes that need to be retained and how they need to be moved based on the VRSs you defined. For detailed information about how VRSs work, see Chapter 11, “Policy management” on page 369.

- **Expiration processing (EXPROC)**

  This processing is used to expire volumes to scratch status according to the management policy. DFSMSrmm calls the EDGUX200 installation exit when running the expiration processing. This exit is called every time that a volume is identified as ready to return to scratch. During expiration processing, DFSMSrmm performs these tasks:

  – Checks the expiration date to identify volumes that are ready to expire.

  – Determines the type of release action that is needed for the volumes that have reached their expiration date.

  – Manages the release actions for volumes before returning them to the scratch pool or removing them from the library.

  – Removes the catalog entry and uncatalogs all parts of the data set across all volumes in the set for a multivolume data set.

  – Optionally updates RACF tape security profiles for volumes returning to scratch status.

  – If UNCATALOG(Y or S) is specified in parmlib member EDGRMMxx, DFSMSrmm uncatalogs all data sets for volumes returning to scratch status.

  – When the EDGRMMxx parmlib option RETAINBY(SET) is in use, DFSMSrmm ensures that all volumes in a set are not expired if any one volume does not expire.

- **Storage location management (DSTORE)**

  Requests storage location management to set the destination for the volume and optionally assigns the exact shelf location to be used. Storage location management processing assigns volume destinations and sets required volume destinations based on the results of other inventory management functions.

  You can select to perform DSTORE by location, specifying the EDGHSKP parameter as shown here:

  `DSTORE(LOCATION(`*`from_location:to_location`*`,...))`

  Note the following explanation:

  – *from_location:to_location* indicates a pair of location names separated by a colon. The *from_location* is the current location from which a volume moves.

  – *to_location* is the name of the required location to which a volume will move. If you omit *to_location*, DFSMSrmm uses “*” as the default.

  You can specify one to eight pairs of *from_location:to_location* names. DSTORE processing is performed then for a volume, if it matches one of the specified location pairs.

  The *from_location* and *to_location* names can be specified in one of the following ways:

  – **Specific**: A specific location name is one to eight characters. The location names that you specify are not validated against the DFSMSrmm LOCDEF entries or the name of system-managed storage (SMS) libraries. For example, SHELF means that just this one location is included.

  – **Generic**: A generic location name is one to seven characters followed by any generic characters that are valid in a single data set qualifier:

    - ATL* means that all location names starting with the characters ATL are included.
    - An asterisk (*) means that all locations are included.

- The notation ABC%%% refers to all locations starting with ABC and with any other character up to a total of a six-character name.

- **Report extract (RPTEXT)**

  When running inventory management, you can choose to request an extract data set, which can be used to create various reports, such as volume movement or inventory.

- **Trial run (VERIFY)**

  Trial run vital record processing does not change data set and volume information in the DFSMSrmm CDS. Use trial run vital record processing to analyze the effect that your movement and retention policies have.

To run inventory management functions, sampleIBM Tivoli Workload Scheduler for z/OS jobs EDGJDHKP and EDGJWHKP are provided in SYS1.SAMPLIB:

- EDGJDHKP (daily job): Invokes the following procedures:
  - EDGPHKP: Runs the EDGHSKP program with PARM= 'VRSEL,EXPROC,RPTEXT'.
  - EDGPMSGC, EDGPVRSA, and EDGPRPTA: Allocate other needed files.
- EDGJWHKP (weekly job): Invokes the following procedures:
  - EDGPHKP: Runs the EDGHSKP program with PARM='DSTORE,VRSEL,EXPROC,RPTEXT'.
  - EDGPMSGC, EDGPVRSA, and EDGPRPTA: Allocate other needed files.

7.1.7 Erasing and labeling volumes

This step automates the labeling and erasing of tape volumes. Sample IBM Tivoli Workload Scheduler for z/OS job EDGJINERS is provided in SYS1.SAMPLIB. DFSMSrmm searches for volumes with INITIALIZE or ERASE actions pending and labels them after optionally erasing them. The job invokes the EDGPINER procedure, which runs the EDGINERS utility with the required parameters.

For detailed information about erasing and initializing tape volumes, see 7.5, "Initializing and erasing volumes" on page 245.

7.1.8 Scratch processing

This step is needed to process volumes that are returning to scratch status. A sample IBM Tivoli Workload Scheduler for z/OS job is available in SYS1.SAMPLIB member EDGJEXP. This job invokes the EDGPEXP procedure, which runs EDGHSKP with the EXPROC parameter to perform expiration processing. It also allocates some other needed files.

For more information about the expiration processing that is performed as part of the EXPROC, see 7.1.6, "Inventory management" on page 215.

7.1.9 Scratch reporting

Perform this step when you need to obtain information about volumes in scratch status. The sample job EDGJSCRL is provided in SYS1.SAMPLIB.

This job invokes the following procedures:

- EDGPRPTX: Creates an updated DFSMSrmm report extract file. It runs the EDGHSKP utility with the RPTEXT parameter. It allocates the new report extract file using the procedure EDGPRPTA.

- EDGPSCRL: It uses the EDGRPTD reporting utility for generating the latest scratch lists using files SCRLIST and NEWSCR.

7.1.10 Ejecting volumes from a system-managed tape library

This task ejects volumes from system-managed tape libraries when volume movement is required. Also, consider VTS export processing in this step. Similar processing is required for any non-system-managed volumes that you have in your library.

A sample IBM Tivoli Workload Scheduler for z/OS job is available in SYS1.SAMPLIB member EDGJEJC. This job invokes the EDGPEJC procedure. You must execute this procedure once for each library in your installation, changing the library name (FROM), destination (TO), and output station (EJECT) in each case.

The EDGPEJC procedure requires two steps:

1. Create a list of volumes to be moved from a system-managed library. The list is created using the command shown in Figure 7-2.

```
RMM SV VOL(*) OWN(*) LIMIT(*) LOC(&FROM) DEST(&TO) INTRANSIT(N) -
               CLIST(“RMM CV”,”EJECT(&EJECT)”)
```

Figure 7-2 RMM SEARCHVOLUME subcommand to create eject commands

   The parameter CLIST allows you to create a set of commands for all the volumes in the generated list.

2. Figure 7-3 shows the commands generated in the previous step for ejecting the volumes.

```
  RMM CV volser EJECT(&EJECT)
  RMM CV volser EJECT(&EJECT)
  ....
```

Figure 7-3 RMM CHANGEVOLUME subcommand to eject the volumes

7.1.11 Producing reports

When running inventory management, you can choose to request an extract data set that can be used to create various reports, such as volume movement or inventory. The extract data set contains information from the CDS. You can use the extract data set as input to EDGRPTD reporting utility, the EDGRRPTE exec, and the report generator to create reports. The extract data set might contain extended extract records.

See *DFSMSrmm Reporting*, SC23-6875, for information about using EDGRPTD and EDGRRPTE.

This step is used to generate movement reports. A sample IBM Tivoli Workload Scheduler for z/OS job is available in the SYS1.SAMPLIB member EDGJMOVE. It invokes the following procedures:

- EDGPRPTX: Creates an updated report extract file, running the EDGHSKP utility with the RPTEXT parameter.
- EDGPMOVE: Uses the updated report extract file as input to the EDGRPTD reporting utility to create movement reports.

The following reports are produced:

- **RDYTOSCR**: Volumes to be moved from locations to the home location
- **TOSTRCK**: Report sorted by rack number including volumes to be moved:
  - From SHELF to storage locations
  - Between system-managed libraries
  - From system-managed libraries to storage locations
- **TOSTOWN**: Report sorted by owner including volumes to be moved:
  - From SHELF to storage locations
  - Between system-managed libraries
  - From system-managed libraries to storage locations
- **FMSTBIN**: Report sorted by bin number including volumes to be moved:
  - From storage location to SHELF
  - Between storage locations
  - From storage locations to system-managed libraries
- **FMSTOWN**: Report sorted by owner including volumes to be moved:
  - From storage location to SHELF
  - Between storage locations
  - From storage locations to system-managed libraries

7.1.12 Move confirmation

Confirmation is needed for volume movement to locations outside the tape library. After the inventory management processing, such as vital record processing and the storage location management processing, DFSMSrmm needs your confirmation about volume movement.

A sample IBM Tivoli Workload Scheduler for z/OS job, EDGJCMOV, is provided in SAMPLIB for performing the move confirmation actions. A global confirmation command is issued, which means that the confirmation is for multiple volumes. The actual confirmation of movement for each volume is done in the next inventory management run.

The procedure EDGPCMOV is invoked, which issues the global confirm move command as shown in Figure 7-4.

```
RMM CV * CMOVE(&FROM,&TO,&TYPE)
```

Figure 7-4 RMM CHANGEVOLUME subcommand to confirm outstanding moves

Provide valid location names for your installation in the parameters FROM, TO, and TYPE, or use ALL to confirm all outstanding moves.

7.1.13 Additional considerations

There are additional considerations for inventory management-related tasks.

Managing tape with permanent errors

DFSMSrmm tracks the permanent I/O errors for a volume, and marks it as needing to be replaced if it is set with a return to scratch release action. DFSMSrmm does not identify those volumes that need the “return to owner” release action.

You can use the RMM SEARCHVOLUME subcommand, the extract data set, or the DFSMSrmm ISPF dialog to check for volumes with temporary errors.

Managing open data sets

In the MESSAGE file, DFSMSrmm lists tape data sets that might be open, as well as messages describing the expiration processing.

These open data sets might be produced by jobs that are currently active. However, some of them also might result from a job failure, meaning the data set was never closed. Review this list and make the necessary corrections. During vital record processing, DFSMSrmm matches open data sets to VRSs using the reserved data set name mask OPEN, or to VRSs defined with data set name masks.

Other considerations

You can perform some activities only under specific conditions:

- If you use the system-managed tape library in your environment, and there are multiple tape configuration databases (TCDBs), you must run inventory management expiration processing at least once against each TCDB.
- When you want to retain data sets based on the catalog information, or if you specify the UNCATALOG option with Y or S in EDGRMMxx to indicate that catalog processing is required, you must run inventory management on a system that has access to the catalogs, or you have to enable catalog synchronization.
- In an unshared user catalog environment, specify the OPTION command with the CATSYSID operand in EDGRMMxx to use DFSMSrmm. You must ensure CDS and user catalogs are synchronized and run expiration processing on each different catalog environment.
- The DFSMSrmm subsystem address space performs all inventory management functions, except backing up the CDS. It is EDGHSKP that backs up the DFSMSrmm CDS.
- You can continue using RMM TSO subcommands and DFSMSrmm continues to record tape usage, while inventory management functions are running.
- At least one VRS must be defined in order to run DFSMSrmm vital record processing. The EDGRMMxx PARMLIB OPTION command VRSMIN operand provides control over this processing.

7.2 Setting up DFSMSrmm common time support

Before DFSMSrmm common time support or Coordinated Universal Time (UTC) is enabled, all dates and times are stored in the DFSMSrmm control data set in local time. When the control data set is shared, and the sharing systems are set to run in different time zones, the local dates and times in the control data set might be from any of your systems. When you display information or extract records, you need to be aware of how the records were created, on which system, and where they might have been updated in order to interpret the dates and times shown. The same consideration also applies for records that are created or updated before enabling common time support because DFSMSrmm assumes that they are times local to the system running the DFSMSrmm subsystem and converts the values based on that assumption.

When you enable common time support, DFSMSrmm maintains the records in the control data set in common time. Most date and time fields are paired together to enable an accurate conversion to and from common time and between different time zones. In some cases, DFSMSrmm has date fields in control data set records, and there is no associated time field. For these date fields, DFSMSrmm uses an internal algorithm that approximates conversion between time zones based on the time zone offsets involved.

Warning: Using the SET system command with either the DATE or the TIME keyword, or both, or replying to message IEA888A to run the system on future or past dates can affect the way that DFSMSrmm calculates local times. In order to get the correct results from DFSMSrmm processing when you need to test with future or past dates, alter the time of day (TOD) clock and keep the time zone offset as before.

7.2.1 How DFSMSrmm uses the date and time

Dates and times in DFSMSrmm are used mostly for reference, but also to calculate retention periods. Here are some examples:

Journal record prefix
: Contains the date and time that the journal records were written, compared for forward recovery

Record creation date and time
: In any record stored in the DFSMSrmm control data set

Expiration date
: In volume and data set records; shows the date that the entity is expired, but not the time

Tokens
: In control data set records; used for various purposes but usually as a unique value to associate multi-volumes and multi-files, or logical volumes and stacked volumes together

Reference dates
: In control data set records

Note: Dates and times are always displayed to the user in the same format that they were recorded, in local time.

All current uses of date and time in DFSMSrmm use the 'TIME DEC' option of the TIME macro; this retrieves the local date and time. DFSMSrmm assumes that all systems that share a CDS and therefore also the JOURNAL run on a synchronized time source.

Although in theory there is supposed to be a common date/time function available in DFSMSrmm and used for all date conversions, in practice it is used only when a date conversion is required or to get the current date in DFSMSrmm internal format. It is not used where the time is wanted or when the date is wanted not in internal format.

DFSMSrmm uses an internal date format that is derived via TIME DEC from the TOD: a 4-byte, 7-character packed decimal number *YYYYDDD*s. Using this value, DFSMSrmm has no known limit to the dates that can be handled. The internal time is a 7-character packed decimal number also, *HHMMSST*s. DFSMSrmm does not need to upgrade to using a 128-bit timer value, or does it have any problems with the TOD clock rolling over in 2043.

7.2.2 Date and time in a DFSMSrmm client/server environment

All dates and times that are used on the client system are in local time. All dates and times that are used on the server are in local time. When data is retrieved from the DFSMSrmm control data set (CDS) and displayed by subcommand processing, the dates and times used are exactly as they are stored in the DFSMSrmm CDS; no conversion from server time zone to client time zone is performed. Figure 7-5 on page 222 shows an example of when you need to use the DFSMSrmm common time support.

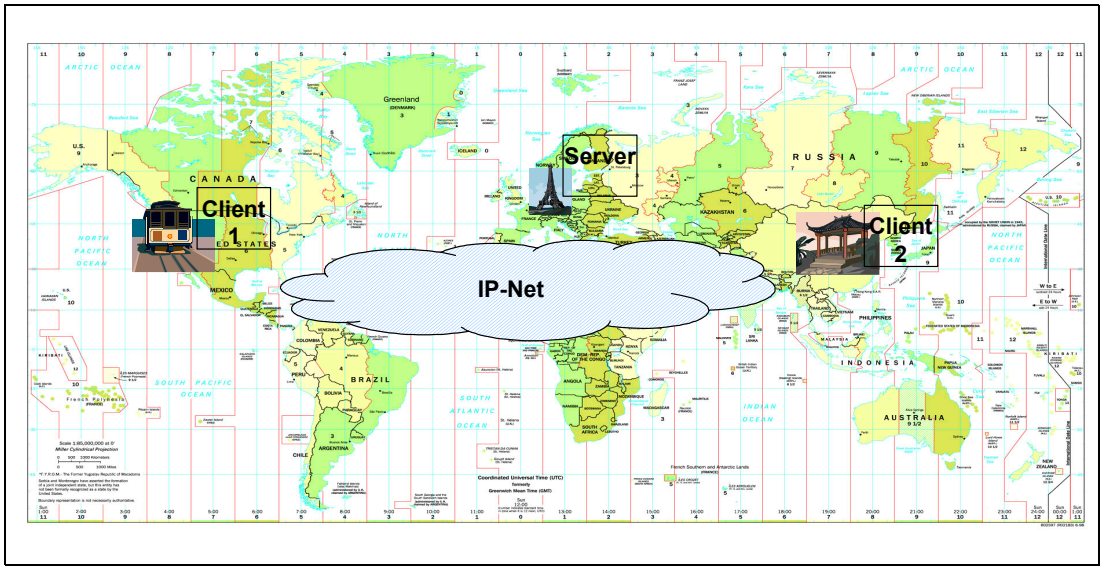


Figure 7-5 DFSMSrmm client/server implementation

7.2.3 Enable common time support

Before you can enable the DFSMSrmm common time support, you must check that the TOD clocks of all systems in the RMMplex are set to Coordinated Universal Time (UTC).

To enable common time support, perform the following tasks:

1. Ensure that all systems in the RMMplex have toleration maintenance installed or are at z/OS V1R8 or higher. Ensure that all applications dependent on the correct date and time information from DFSMSrmm are updated to support the new time zone support, if required. The DFSMSrmm subcommand output remains in local time, so most applications do not need to change unless they are to use the availability of the time zone offset.
2. Ensure that the system TOD clock is set to Greenwich mean time (GMT) on all systems in the RMMplex. It is common practice for the system to use local time based either on the TIMEZONE value in the CLOCK*xx* member of parmlib or from an external time source.
3. Run the EDGUTIL utility with UPDATE with the UTC(YES) operand on the CONTROL statement of the SYSIN file to enable common time support.

Important: You cannot disable common time support once it is enabled.

UTC(YES) enables DFSMSrmm common time support. Before you enable this support, all systems in the RMMplex must have toleration maintenance installed or be at z/OS V1R8 or higher. Applications that are dependent on the correct date and time information from DFSMSrmm must be updated to support the new time zone support if required.

Example 7-1 shows the result of an RMM TSO LISTCONTROL command if you do not have DFSMSrmm common time support enabled.

Example 7-1 Common time support disabled

```
Control record:
Type = MASTER    Create date = 24/04/1997  Create time = 14:28:32
                 Update date = 01/11/2012  Update time = 20:03:26
Journal: Utilization =   3% (75% threshold)     STATUS: = ENABLED
```

```
CDS:     Utilization =  52%
Exit status:                                Options:
  EDG_EXIT100 = ENABLED                       Stacked Volumes         = DISABLED
  EDG_EXIT200 = NONE                          Extended Bin            = DISABLED
  EDG_EXIT300 = NONE                          Common Time             = DISABLED
                                              CDSID ENQ name          = ENABLED
Last backup:                                Last expiration processing:
  Date = 01/11/2012  Time = 07:00:03          Date = 01/11/2012  Time = 07:00:06
Last journal backup:                        Last store update:
  Date = 01/11/2012  Time = 07:00:03          Date = 30/10/2012  Time = 07:01:45
Last report extract:                        Last VRS processing:
  Date = 01/11/2012  Time = 07:01:48          Date = 01/11/2012  Time = 07:00:06
Last scratch procedure:                     Last Catalog synchronize:
  Date = 17/02/2010  Time = 12:22:11          Date = 11/09/2006  Time = 08:29:18
Rack numbers       = 8                      Empty racks        = 8
LOCAL store bins   = 243                    Empty LOCAL bins   = 243
DISTANT store bins = 300                    Empty DISTANT bins = 269
REMOTE store bins  = 999                    Empty REMOTE bins  = 999
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =
```

Use Example 7-2 to enable DFSMSrmm common time support.

Example 7-2 Enable UTC support

```
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
 CONTROL UTC(YES) CDSID(SC70)
/*
```

The following descriptions apply to Example 7-2:

CDSID Specifies one-to-eight alphanumeric characters that identify the control data set by name. There is no default. A CDSID is required in z/OS 1.9 or higher.

UTC Enables DFSMSrmm common time support.

When you enable common time support, DFSMSrmm starts to record CDS record dates and times in common time and converts existing values, as required, from local times to common time. Figure 7-6 on page 224 shows an example of a display data set.

Each user can set their own common time zone values to display the date and time information correctly.

```
  Panel  Help
 ____________________________________________________________________________
EDGP§OP1                 DFSMSrmm Dialog User Options
Command ===>

Date format . . . . . JULIAN          ( American, European, Iso or Julian )
Time zone . . . . . . MST   -04:00:00  ( zone  offsetHH:MM:SS )

Confirm deletes  . . . NO               ( Yes or No )

Processing option  . . F                 F - Foreground, B - Background
DSNAME case option . . U                 M - Mixed, U - Upper
Eject option . . . . . C                 C - Convenience, B - Bulk

Variable reuse . . . . Y                 Y - Yes, N - No

Job statement information:-

 ===> //RMMJOB   JOB ,RMM,NOTIFY=&SYSUID,
 ===> //         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
 ===> //*
 ===> //*

      Enter END command to save changes, or CANCEL to end without saving.
```

Figure 7-6 Set your own common time zone value

All dates and times that are displayed or entered in the DFSMSrmm dialog are values local to this time zone. To change the time zone, you must specify an offset value and a text string to identify that zone to you. The offset value is the amount of time that your selected time zone is ahead of or behind universal time (UTC). Changes you make only affect future dialog actions and displays.

The report extract data set contains date and time values in the local time of the running system. The extract header record includes a field that lists the time zone offset as shown in Figure 7-7.

```
H          2007/072  183039SC64                                    JN-04:00:00
```

Figure 7-7 Report extract data set time zone offset

Suggestion: We suggest that you have the system Time Of Day (TOD) clock set to GMT and enable DFSMSrmm to use UTC. You do this by using the EDGUTIL utility with the UPDATE parameter. Once you do this, any newly recorded dates and time will be stored in common time and existing records will be converted to be in all common time as they are updated. You will continue to see dates and times presented in local time because DFSMSrmm handles the conversion from common time to your local time.

Set the TIMEZONE in the CLOCKxx PARMLIB member

TIMEZONE *d.hh.mm.ss* specifies the difference between the local time and the Coordinated Universal Time (UTC). If ETRMODE YES and ETRZONE YES are specified (and an operational Sysplex Timer is available), the system ignores the TIMEZONE parameter.

The following descriptions apply to TIMEZONE d.hh.mm.ss:

d Specifies the direction from UTC. Value range: E for east of UTC or W for west of UTC. Default: W

hh.mm.ss Specifies the number of hours (*hh*) minutes (*mm*) and seconds (*ss*) that the local time differs from the UTC. Value range: The value for *hh* must be between 00 and 15. The value for *mm* and *ss* must be between 00 and 59. *mm.ss* values are optional. In addition, the combined *hh.mm.ss* value must be within the range (00:00:00 - 15:00:00). This means that a value, such as 15.59.59, is not valid because it is outside the range, even though the *hh* portion is between 00 and 15 and the *mm* and *ss* portions are between 00 and 59. If the *mm* portion or *ss* portion or both are omitted, a default value of 00 is applied to the omitted portion, and appears in message IEA598I at IPL time. For example, if CLOCKxx contains TIMEZONE W.15, at IPL time, message IEA598I indicates: `IEA598I TIME ZONE = W.15.00.00` Default: 00.00.00

Example 7-3 shows how you can use the DISPLAY T command to display the local time of day and date and the Coordinated Universal Time (UTC) of day and date.

Example 7-3 Display the Local and Coordinated Universal Time and Date

```
D T
```

The local time of day and date and the Coordinated Universal Time (UTC) of day and date are to be displayed (message IEE136I) as shown in Figure 7-8.

```
RESPONSE=SC70      IEE136I LOCAL: TIME=12.56.19 DATE=2007.068  UTC:
RESPONSE=TIME=17.56.19 DATE=2007.068
```

Figure 7-8 Result of displaying the Local and Coordinated Universal Time and Date

7.2.4 Potential problems using local time

The following problems might occur when using local time:

- Systems sharing the CDS but in different time zones create records using their own local date and time values.
- Journal records might appear to not be in sequence; however, the VSAM shared information (VSI) count and record numbers show the correct sequence. Tokens vary but always increment.
- Users see a date and time and believe it to be in their local time zone, but it can reflect the date and time that is used on another system.
- Data sets or volumes can be released early.

7.3 Dynamic installation exits

DFSMSrmm now uses the system Dynamic Exit Services to manage calls to the installation exits, determine whether exit modules exist, and to provide error handling and recovery. The installation exits that DFSMSrmm provides are shown in Table 7-2 on page 226.

Table 7-2 Sample installation exits IBM provides

Exit name	Default exit routine	SAMPLIB samples provided
EDG_EXIT100	EDGUX100	EDGUX100, EDGCVRSX
EDG_EXIT200	EDGUX200	EDGUX200
EDG_EXIT300	EDGUX300	EDGUX300

DFSMSrmm installation exit modules can be loaded from any authorized program facility (APF)-authorized library. The default is the LINKLST.

Dynamic Exit Services is used to load and activate the default (EDGUXn00) exit module at initialization time. The default exit is activated in first position. All other exit-related processing is handled by Dynamic Exit Services. If you do not use the default exit module (EDGUXn00), you must use PROGxx in z/OS parmlib, the SETPROG operator command, or the CSVDYNEX macro.

Important: When the definition of the dynamic exits or activation of the default exits encounters a problem, DFSMSrmm start-up provides the operator with a choice of actions: CANCEL, RETRY, or CONTINUE.

7.3.1 DFSMSrmm installation exit calls

There are several exit points in Allocation and Open Processing where exit module EDGUX100 gets called to support functions, such as “Special Date” or “Sticky Label”. During inventory management, EDGUX200 gets called before a volume returns to scratch. At one exit point during Open, EDGUX300 provides information about the tape media. See Figure 7-9 to understand how DFSMSrmm calls the exit.

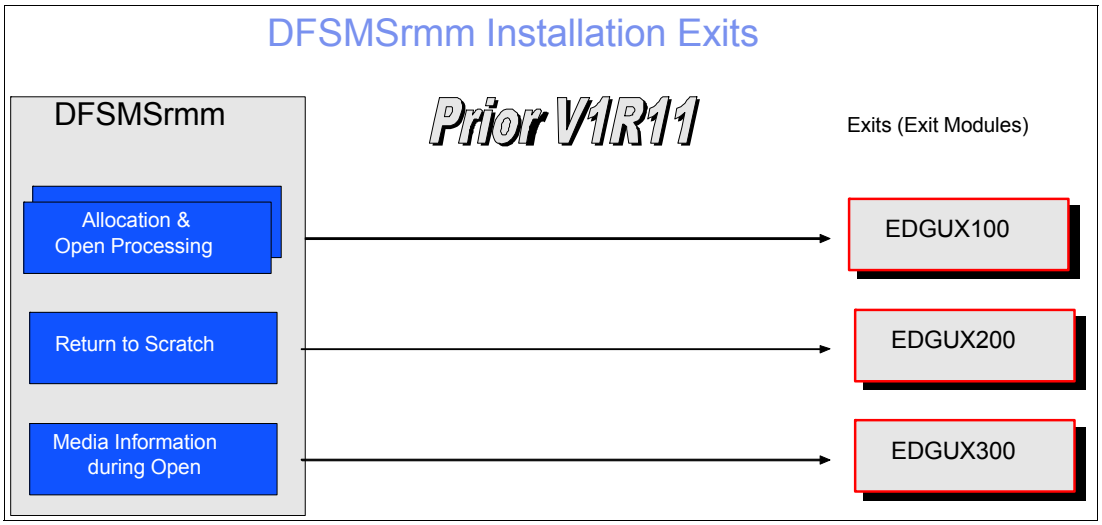


Figure 7-9 Prior z/OS 1.11 DFSMSrmm installation exit call

Here is a short explanation of what things have and have not changed in z/OS V1.11 with dynamic exits. The blue side on the left of Figure 7-9 shows that the exit points in DFSMSrmm are still the same. Also, the sample exit modules on the right, (EDGUX100/200/300), which are shipped with DFSMSrmm, have not been changed. The parameter lists that are mapped by EDGPL100/200/300 have not been changed either.

Figure 7-10 shows the three new exits in the middle (EDG_EXIT100/200/300), which are controlled by Dynamic Exit Services. DFSMSrmm no longer calls the exit modules directly. Also new is the ability to add more exit modules to a given exit.

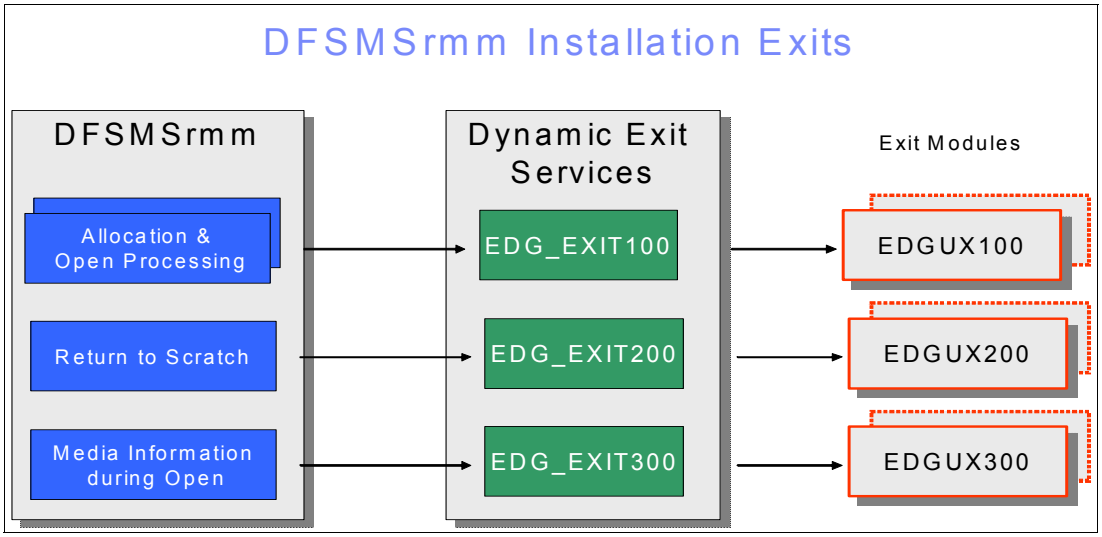


Figure 7-10 z/OS 1.11 DFSMSrmm installation exit call

Note: Although the exit points have not changed, error handling has changed.

7.3.2 Managing exit modules

Before z/OS V1.11, the exit modules were not able to be managed independently. Also, it was not possible to temporarily deactivate an exit module. Instead, the exit module had to be deleted from the LINKLIB before the REFRESH EXITS command was issued. Figure 7-11 shows the use of the DFSMSrmm REFRESH EXITS command. There is no capability to specify an exit in the command to refresh only one exit.

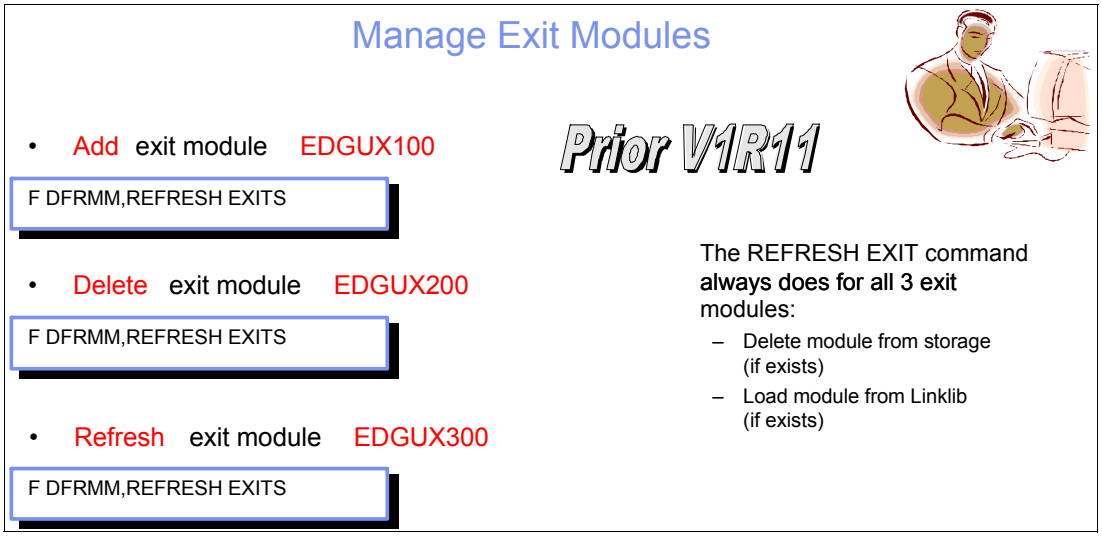


Figure 7-11 Managing exit modules prior z/OS 1.11

With z/OS V1.11, the MODIFY DFRMM,REFRESH EXIT command is no longer supported. You can use the SETPROG EXIT command instead. This command allows you to manage the exit modules independently. Remember that the EXIT statement of the PROGxx parmlib member can also be used to add an exit module. In Figure 7-12, you can see how you can refresh a single DFSMSrmm installation exit using the SETPROG operator command.

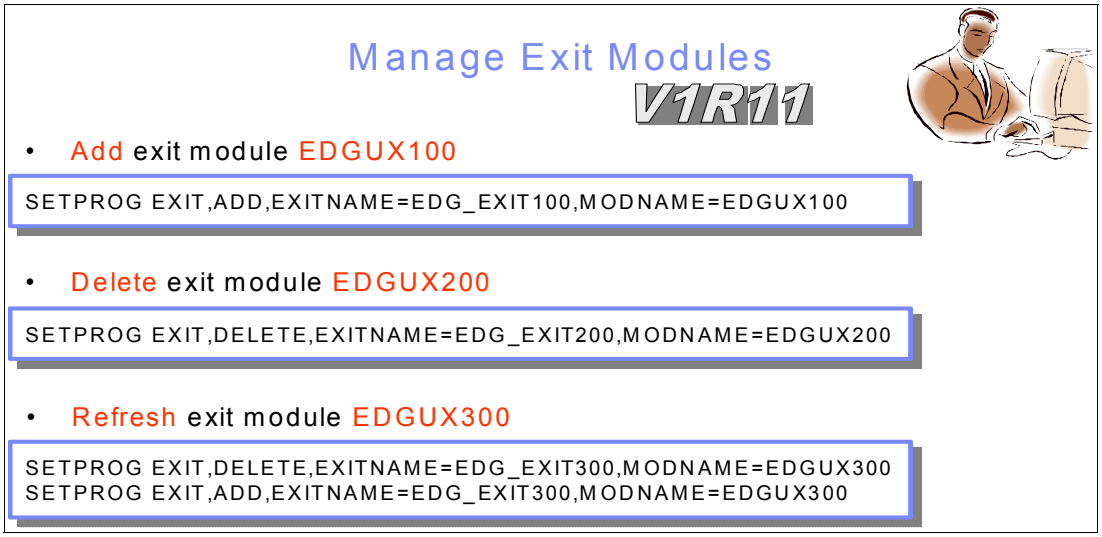


Figure 7-12 Managing exit modules with z/OS 1.11

Important: If you have installed DFSMSrmm z/OS 1.11 or lower, quiesce DFSMSrmm at this time by using the following command:

```
MODIFY DFRMM,QUIESCE
```

After the exits are refreshed, you must restart DFSMSrmm. You can also use the following command:

```
SETPROG EXIT,ADD,EXITNAME=EDGUX100_EXIT,MODNAME=EDGUX101
```

Next, delete the existing EDGUX100:

```
SETPROG EXIT,DEL,EXITNAME=EDGUX100_EXIT,MODNAME=EDGUX100
```

If EDGUX101 is working correctly, copy it back over EDGUX100, and issue the following commands, which will put the RMM environment back to “normal”:

```
SETPROG EXIT,ADD,EXITNAME=EDGUX100_EXIT,MODNAME=EDGUX100
SETPROG EXIT,DEL,EXITNAME=EDGUX100_EXIT,MODNAME=EDGUX101
```

In addition to deleting and adding an exit module, you can also temporarily deactivate it, but keep in mind that setting the state to ACTIVE again does not imply a reload of the module from the library. See Figure 7-13 on page 229.

Manage Exit Modules

V1R11

- Inactivate exit module EDGUX300

```
SETPROG EXIT,MODIFY,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300,STATE=INACTIVE
```

- Activate exit module EDGUX300

```
SETPROG EXIT,MODIFY,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300,STATE=ACTIVE
```

Figure 7-13 Deleting and adding exit modules with z/OS 1.11

Note: Inactivate or activate does not reload the exit module from the library.

For V1R11 and earlier, the exit status was shown for what are now the exit routines EDGUX100, EDGUX200, and EDGUX300. Now, the status is shown for the exits EDG_EXIT100, EDG_EXIT200, and EDG_EXIT300. Nevertheless, the statuses ENABLED, DISABLED, and NONE have not changed. However, the exact descriptions of the statuses have been adjusted to these descriptions:

NONE	Exit is not defined or no exit modules exist.
ENABLED	At least one active exit module exists.
DISABLED	One or more exit modules exist but is none is active.

In Figure 7-14, you can see the status of each installation exit if you are using the TSO RMM LISTCONTROL subcommand.

```
Control record:
Type = MASTER    Create date = 2002/064    Create time = 15:46:09
                 Update date = 2009/229    Update time = 20:05:25
Journal: Utilization =   8% (75% threshold)     STATUS: = ENABLED
CDS:     Utilization =  29%
Exit status:                              Options:
  EDG_EXIT100 = ENABLED                     Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = NONE                        Common Time             = ENABLED
                                            CDSID ENQ name          = ENABLED
```

Figure 7-14 LISTCONTROL output

Figure 7-15 on page 230 shows the same information when you are using the DFSMSrmm ISPF dialog.

```
                        DFSMSrmm Control Record Display
Command ===>

                                                                  More:     +
Type  . . : MASTER      Create date . : 2002/064    Create time . : 15:46:09
                        Update date . : 2009/229    Update time . : 20:05:25

Journal utilization . :   8 %    ( 75 % threshold)  Status : ENABLED
CDS utilization . . . :  29 %

Exit status:                            Options:
EDG_EXIT100 . . : ENABLED                  Stacked volumes . . . : NONE
EDG_EXIT200 . . : NONE                     Extended bin  . . . . : DISABLED
EDG_EXIT300 . . : NONE                     Common time . . . . . : ENABLED
...
```

Figure 7-15 DFSMSrmm control panel display

7.3.3 Setting Up the EDG_EXIT100 routine environment

To activate or reactivate the exit routine, you can use one of the following tasks:

- Use dynamic exit services via operator command.
- Use the CSVDYNEX macro.
- Start DFRMM for the first time since an IPL.

Note: Stopping and restarting, or refreshing, the DFSMSrmm procedure does not reload or reactivate your exit routine. See *DFSMSrmm Managing and Using Removable Media*, SC26-7404, for more information.

To install the EDGUX100 default exit routine, perform these actions to update or replace the default exit module:

1. Build and install an SMP/E USERMOD to apply the updated source code for the EDGUX100 exit module. Include the necessary JCLIN statements to get the EDGUX100 load module added to the LINKLIB target library. You can install the load module by applying the updated source code using an SMP/E USERMOD as shown in Figure 7-16 on page 231. Modify the FMID and PRE to reflect the release you are running:

   a. Allocate a user SAMPLIB data set. In Figure 7-16 on page 231, the user SAMPLIB data set is defined as MY.RMM.SRCLIB and allocated to DD card SRCLIB.

   b. Copy the EDGUX100 source from SAMPLIB to the user SAMPLIB and modify, as needed, for your installation.

   c. SMP/E RECEIVE the USERMOD.

   d. SMP/E APPLY the USERMOD. Ensure that a DD card exists for the user SAMPLIB in the APPLY job, or as a DDDEF to SMP/E in the target zone.

   After performing these steps, the modified version of the EDGUX100 exit module source code resides in both the user SAMPLIB and SYS1.SAMPLIB. The original IBM copy is only in the distribution libraries at this point. If you accept the USERMOD, only the modified version of the sample exit routine source code exists. The SMP/E target zone reflects RMID indicators of VMRMM01 for all of these records:

   – SAMP EDGUX100 RMID=VMRMM01 SYSLIB=SAMPLIB
   – SRC EDGUX100 RMID=VMRMM01 SYSLIB=SAMPLIB
   – MOD EDGUX100 RMID=VMRMM01 LMOD=EDGUX100

– LMOD EDGUX100 SYSLIB=LINKLIB

The RMID of VMRMM01 for the SAMP record prevents IBM service from being installed. This results in an ID search and notification to you that IBM is the servicing exit source code.

```
//STEP1    EXEC SMPEMVS,REGION=6120K
//SYSIN    DD   *
 SET BDY(GLOBAL) -
 RECEIVE
/*
//SMPPTFIN DD DATA,DLM=##
++USERMOD (VMRMM01) REWORK(1992082) .
++VER (Z038) FMID(HDZ11D0) .
++JCLIN .
//EDGUX100 EXEC PGM=IEWL,
//         PARM='LET,NCAL,RENT,REUS,LIST,XREF,REFR'
//SYSLMOD  DD DSN=SYS1.LINKLIB,DISP=SHR
//SRCLIB   DD DSN=MY.RMM.SRCLIB,DISP=SHR
//AEDGMOD1 DD DSN=SYS1.AEDGMOD1,DISP=SHR
//SYSPRINT DD SYSOUT=*
  INCLUDE AEDGMOD1(EDGUX100)
  ENTRY   EDGUX100
  NAME    EDGUX100(R)
++SRC(EDGUX100) TXLIB(SCRLIB) DISTLIB(ASAMPLIB) .
++SAMP(EDGUX100) TXLIB(SCRLIB) DISTLIB(ASAMPLIB) .
##
/*
```

Figure 7-16 Create a SMP/E USERMOD to apply the updated EDGUX100 installation exit

2. Copy the new exit load module into the LNKLST library.
3. Refresh library lookaside (LLA).
4. Refresh the exit by using MVS operator commands.

If DFSMSrmm is running in a shared environment, you must repeat this step on each system.

When you use the default exit module, EDGUX100, for the installation exit EDG_EXIT100, it is loaded and activated as an exit routine, if not already active by DFSMSrmm when DFSMSrmm is started. It stays loaded and active for the life of the IPL. It can be refreshed or deleted at any time by using one of the MVS system operator commands.

You can use any load module name for your exit module because you use PROGxx in z/OS parmlib, SETPROG, or CSVDYNEX macro to associate the exit module with the exit. Therefore, you do not have to use EDGUX100 as the load module name. However, using the default exit module name simplifies your implementation because DFSMSrmm itself ensures that the exit module is loaded and activated. Figure 7-17 shows the correct use of the SETPROG MVS command to add the EDG_EXIT100.

```
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 7-17 SETPROG EXIT ADD syntax

The RMM TSO LISTCONTROL subcommand you can see in Figure 7-18 on page 232 shows the current status of the three possible DFSMSrmm installation exits.

```
RMM LISTCONTROL

 Control record:
 Type = MASTER    Create date = 2002/064    Create time = 15:46:09
                  Update date = 2009/235    Update time = 02:00:19
 Journal: Utilization =  46% (75% threshold)     STATUS: = ENABLED
 CDS:     Utilization =  29%
 Exit status:                            Options:
   EDG_EXIT100 = ENABLED                   Stacked Volumes         = NONE
   EDG_EXIT200 = NONE                      Extended Bin            = DISABLED
   EDG_EXIT300 = ENABLED                   Common Time             = ENABLED
                                           CDSID ENQ name          = ENABLED
...
```

Figure 7-18 RMM TSO LISTCONTROL

7.3.4 Controlling the exit routine through the z/OS dynamic exits facility

IBM has defined the EDG_EXITn00 exit to the dynamic exits facility. You can refer to the exit by the name EDG_EXITn00. You can use the EXIT statement of the PROGxx parmlib member, the SET PROG=xx operator command, the SETPROG EXIT operator command, or the CSVDYNEX macro to control this exit and its exit routines. Figure 7-19 shows a sample of SETPROG to add the EDGUX300 user exit after an IPL.

```
SETPROG EXIT,ADD,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300
```

Figure 7-19 SETPROG EXIT ADD syntax

You can use the ADDABENDNUM and ABENDCONSEC parameters on the CSVDYNEX REQUEST=ADD macro or the ABENDNUM parameter of the SETPROG EXIT operator command to limit the number of times the exit routine abnormally ends before it becomes inactive. An abend is counted when both of the following conditions exist:

- The exit routine does not provide recovery, or the exit routine does provide recovery but percolates the error.
- The system allows a retry; that is, the recovery routine is entered with bit SDWACLUP off.

By default, the system disables the exit routines for EDG_EXIT100 and EDG_EXIT300 after one abend. By default, the system does not disable the exit routines for EDG_EXIT200.

You can use the MVS SETPROG command to change the state of an exit routine. In Figure 7-20, we change the status from ACTIVE to INACTIVE.

```
SETPROG EXIT,MODIFY,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300,STATE=INACTIVE
```

Figure 7-20 SETPROG EXIT MODIFY syntax

After you have deactivated EDG_EXIT300, you can control the result using the RMM TSO LISTCONTROL command as shown in Figure 7-21 on page 233.

```
RMM LISTCONTROL

Control record:
Type = MASTER    Create date = 2002/064    Create time = 15:46:09
                 Update date = 2009/235    Update time = 02:00:19
Journal: Utilization =  46% (75% threshold)     STATUS: = ENABLED
CDS:     Utilization =  29%
Exit status:                              Options:
  EDG_EXIT100 = ENABLED                     Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = DISABLED                    Common Time             = ENABLED
                                            CDSID ENQ name          = ENABLED
...
```

Figure 7-21 RMM TSO LISTCONTROL

7.3.5 Deleting the EDG_EXIT100 exit routines

To delete, or selectively delete, the exit routines for the EDG_EXIT100 exit from the system, you can use MVS operator commands. If DFSMSrmm is running in a shared environment, you must repeat this step on each system. Figure 7-22 shows the correct use of the MVS SETPROG command to delete EDG_EXIT300.

```
SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT300,MODNAME=EDGUX300
```

Figure 7-22 SETPROG EXIT DELETE syntax

Now, the status for EDG_EXIT300 is NONE and no longer DISABLED as shown in Figure 7-23.

```
RMM LISTCONTROL

Control record:
Type = MASTER    Create date = 2002/064    Create time = 15:46:09
                 Update date = 2009/235    Update time = 02:00:19
Journal: Utilization =  46% (75% threshold)     STATUS: = ENABLED
CDS:     Utilization =  29%
Exit status:                              Options:
  EDG_EXIT100 = ENABLED                     Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = NONE                        Common Time             = ENABLED
                                            CDSID ENQ name          = ENABLED
...
```

Figure 7-23 RMM TSO LISTCONTROL

7.3.6 Writing an exit routine for the EDG_EXIT100 exit

EDG_EXIT100 exit routines run in SYSTEM KEY 0 or 5 AMODE(31) RMODE(ANY) in the user's address space. KEY 0 is used when a WTO address or an ACERO address is provided, and KEY 5 is used when a JFCB address is provided. The default exit routine, EXGUX100, is loaded and activated via the CSVDYNEX macro, the systems' dynamic exits services. Exit modules can be contained in any APF-authorized LNKLST library or an APF-authorized library named on CSVDYNEX macro invocation.

7.4 Maintaining your control data set

DFSMSrmm provides three different utilities to maintain your DFSMSrmm CDS. The CDS is a Virtual Storage Access Method (VSAM) key-sequenced data set (KSDS) and needs to be backed up once a day and reorganized from time to time. In this section, we describe how you can back up your CDS and journal, reorganize your CDS, and restore your CDS. The utilities provide some more functions, but these functions are not described in this section. For more information about these utilities, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

The following functions of the DFSMSrmm utilities maintain the CDS that we describe in this section:

EDGHSKP To back up the CDS and journal

EDGBKUP To back up the CDS and journal, reorganize the CDS, or to restore the CDS

EDGUTIL To create or update the CDS control record

7.4.1 Using EDGHSKP

You can use the EDGHSKP utility to back up your DFSMSrmm CDS and journal, and to clear the journal. You do not have to stop any DFSMSrmm function while this utility is running. DFSMSrmm stops updates that are made to the CDS while the access method services (AMS) backup function is running. If you are using DSS to back up the CDS, updates are allowed.

The EDGHSKP utility requests that DFSMSrmm is initialized and the task is started.

BACKUP

Specify BACKUP to back up the control data set to back up the journal, or to back up both the control data set and the journal. When you specify BACKUP on the EXEC statement, you must also specify a DD statement. Specify BACKUP DD when you back up the control data set, and JRNLBKUP DD when you back up the journal. You can also specify both DD statements in your EDGHSKP JCL.

The following descriptions apply:

AMS BACKUP(AMS) is the default. DFSMSrmm uses the access method services REPRO command to perform backup processing. Specify BACKUP(AMS) to prevent DFSMSrmm from updating the control data set during control data set backup processing. Backup cannot be directly to tape.

DSS Specify BACKUP(DSS) with the DFSMSdss concurrent copy environment setup to permit the update of the DFSMSrmm control data set during backup processing. This ensures that all tape activities can continue during backup processing. You can specify BACKUP(DSS) without setting up the concurrent copy environment. If you specify BACKUP(DSS), backup can be direct to tape.

COPY Specify BACKUP(COPY) to use DFSMSdss to create a copy of the CDS and optionally, a backup of the journal. The COPY uses DFSMSdss logical data set copy and enables you to use fast replication services that enable CDS updates to be made while the copy is created. COPY works just as the DSS option, but uses the COPY command instead of the DUMP command with DFSMSdss.

The BACKUP DD is optional for COPY. Specify it if you want to direct the copy to a specific volume or if your environment is not system-managed storage (SMS)-managed.

We suggest that you use DFSMSrmm utilities to back up the CDS and journal data set instead of other backup tools. You can plan a batch job using EDGHSKP or EDGBKUP for backup, or set up an automatic backup procedure, as well.

You can run DFSMSrmm control data set backup processing using the DFSMSdss DUMP command, or the access method services REPRO command.

When you use DFSMSdss concurrent copy for backup, DFSMSrmm continues to process commands and update the control data set.

When you use the access method services REPRO command or DFSMSdss without concurrent copy for backup, DFSMSrmm does not allow updates to the control data set during control data set backup.

Certain DFSMSrmm requests wait until the backup completes. RMM TSO subcommands that request to update the DFSMSrmm control data set fail. Only actions that read the control data set continue uninterrupted.

To use DFSMSdss concurrent copy, you must authorize the DFSMSrmm batch inventory management backup job to the RACF profile STGADMIN.ADR.DUMP.CNCURRNT.

For journal backup, DFSMSrmm uses IDCAMS to back up the journal even when BACKUP(DSS) is specified. When using EDGHSKP to perform backup, the journal can be cleared.

Note: When DFSMSrmm is active, you do not need to specify the MASTER and JOURNAL DD statements. EDGBKUP obtains the name of the control data set and the journal from the DFSMSrmm subsystem. When the DFSMSrmm subsystem is stopped, quiesced, or when you want to back up a control data set that is not in use by DFSMSrmm, specify the MASTER DD statement and the JOURNAL DD statement.

Example 7-4 shows EDGHSKP JCL using access method services to perform backup processing. The backup copies must be allocated on DASD.

Example 7-4 CDS and journal backup using AMS

```
//CDSHKUP  JOB ....
//STEP0001 EXEC PGM=EDGHSKP,PARM='BACKUP'
//MESSAGE  DD   DISP=SHR,DSN=RMM.MESSAGE.FILE
//SYSPRINT DD   SYSOUT=*
//BACKUP   DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9216
//JRNLBKUP DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.JRNLBKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9248
```

Note: When you use AMS, the backup copy must be allocated on DISK, because until the backup processes, the DFSMSrmm control data set (CDS) is locked and it is not possible to open a tape data set.

The following explanations apply to Example 7-4 on page 235:

BACKUP DD Specifies to back up the control data set
JRNLBKUP DD Specifies to back up the journal
MESSAGE DD Lists the messages the DFSMSrmm subsystem issues
SYSPRINT DD Contains the utility program messages

If you specify both DD statements to back up the CDS and journal, and the function ended without any errors, the journal is cleared at the end.

Example 7-5 shows a sample EDGHSKP JCL using DFSMSdss to perform backup processing. In this case, the output data set can be allocated on TAPE, but in a case of recovery, the actual journal and the latest backup copy of the journal are needed.

Example 7-5 CDS and journal backup using DFSMSdss

```
//STEP0001 EXEC PGM=EDGHSKP,PARM='BACKUP(DSS)
//MESSAGE  DD   DISP=SHR,DSN=RMM.MESSAGE.FILE
//SYSPRINT DD   SYSOUT=*
//BACKUP   DD   DISP=(,CATLG),UNIT=TAPE,
//         LABEL=(1,SL),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9216
//JRNLBKUP DD   DISP=(,CATLG),UNIT=TAPE,
//         LABEL=(2,SL),VOL=REF=*.BACKUP,
//         DSN=RMM.HSKP.JRNLBKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9248
```

The following explanations apply to Example 7-5:

BACKUP DD Specifies to back up the control data set
JRNLBKUP DD Specifies to back up the journal
MESSAGE DD Lists the messages the DFSMSrmm subsystem issues
SYSPRINT DD Contains the utility program messages

If you specify both DD statements to back up the CDS and journal, and the function ends without any errors, the journal is cleared at the end.

Optionally, you can specify a DSSOPT DD statement to customize the DFSMSdss DUMP or COPY command.

To set up automatic backup for CDS, follow these steps.

Set the OPTION command in EDGRMMxx:

- Specify the JOURNALFULL value. The default is 75%, and a zero value means disable the function.
- Specify the name of the backup procedure using the BACKUPPROC operand.
- Create a backup procedure in PROCLIB.
- For the OPTION command in EDGRMMxx, see Figure 7-24.

```
OPTION ...
          BACKUPPROC(EDGBKUP)              /* Name of BACKUP-proc      */-
          JOURNALFULL(75)                  /* Percentage JRNL full     */-
        ...
```

Figure 7-24 PARMLIB setting for automatic backup

Example 7-6 shows a sample backup procedure.

Example 7-6 Procedure for backing up CDS and journal

```
//EDGBKUP  PROC OPT=BACKUP(AMS)
//STEP0001 EXEC PGM=EDGHSKP,PARM='&OPT'
//MESSAGE  DD   DISP=SHR,DSN=RMM.MESSAGE.FILE
//SYSPRINT DD   SYSOUT=*
//BACKUP   DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,LRECL=9216,BLKSIZE=0,BUFNO=30
//JRNLBKUP DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.JRNLBKUP(+1),
//         RECFM=VB,LRECL=9248,BLKSIZE=0,BUFNO=30
```

Note: When the inventory management task is running, the DFSMSrmm does not perform automatic backup. If multiple DFSMSrmm subsystems share CDS, running automatic backup on only one system is better, or define different JOURNALFULL values.

7.4.2 Using EDGBKUP

You can use the EDGBKUP utility to reorganize and restore your DFSMSrmm CDS. Also, you can use this utility to back up your DFSMSrmm CDS and journal, and to clear the journal. You have to stop DFSMSrmm or use the QUIESCE command to quiesce the DFSMSrmm subsystem interface. For more information about QUIESCE, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

The EDGBKUP utility can be used if DFSMSrmm is stopped or quiesced.

BACKUP

Specify BACKUP to back up the control data set, to back up the journal, or to back up both the control data set and the journal. When you specify BACKUP on the EXEC statement, you must also specify a DD statement. Specify BACKUP DD when you back up the control data set, and JRNLBKUP DD when you back up the journal. You can also specify both DD statements in your EDGHSKP JCL. The following explanations apply to BACKUP:

COPY Specify BACKUP(COPY) to use DFSMSdss to create a logical data set copy of the control data set and, optionally, a backup of the journal. DFSMSrmm uses DFSMSdss to create the copy of the control data set and IDCAMS to back up the journal. To allow updates to the control data set during backup processing, set up the DFSMSdss concurrent copy environment, virtual concurrent copy environment, or a fast replication with virtual concurrent copy environment. You must have the hardware and software that are required to establish a concurrent copy session, a virtual concurrent copy session, or fast replication with concurrent copy or virtual concurrent copy.

DSS Specify BACKUP(DSS) to use DFSMSdss to back up the control data set. DFSMSrmm uses DFSMSdss to back up the control data set and IDCAMS to back up the journal. To allow updates to the control data set during backup processing, set up the DFSMSdss concurrent copy environment or virtual concurrent copy environment. You must have the hardware and software that are required to establish a concurrent copy session or a virtual concurrent copy session.

NREORG Specify BACKUP(NREORG) to use IDCAMS to back up the control data set and to back up the journal data set.

NREORG is the default.

For journal backup, DFSMSrmm uses IDCAMS to back up the journal even when BACKUP(DSS) is specified. When using EDGHSKP to perform backup, the journal can be cleared.

Example 7-7 shows sample EDGBKUP JCL. DFSMSrmm is stopped or quiesced using the access method services to perform backup processing.

Example 7-7 CDS and journal backing up using EDGBKUP with option AMS

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='BACKUP'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//JOURNAL  DD   DISP=SHR,DSN=RMM.PROD.JRNL
//BACKUP   DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9216
//JRNLBKUP DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.JRNLBKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9248
```

The following descriptions apply to Example 7-7:

BACKUP DD Specifies to back up the control data set

JOURNAL DD Identifies the DFSMSrmm journal to be used during backup processing

JRNLBKUP DD If you want to back up the journal

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

If you specify both DD statements to back up the CDS and journal, and the function ends without any errors, the journal is cleared at the end.

Note: If DFSMSrmm is stopped or quiesced, code the MASTER and JOURNAL DD if you have requested to back up these files.

REORG

Specify BACKUP(REORG) to reorganize the control data set during backup processing. DFSMSrmm uses IDCAMS to back up the control data set, to optionally back up the journal, and to restore the control data set from the backup.

The DFSMSrmm subsystem must be stopped or quiesced when you reorganize the active control data set. DFSMSrmm reorganizes the control data set by first backing up the control data set and optionally the journal. Then, DFSMSrmm restores the control data set from the backup control data set.

Note: Do not use the BACKUP(DSS) parameter to reorganize the control data set.

Before you start the EDGBKUP utility to reorganize the control data set, you have to stop or quiesce the DFSMSrmm subsystem. In our example, we only quiesce the DFSMSrmm subsystem. If you use quiesce, you do not have to stop all your tape activities, nor do you have to switch your tape drives offline.

To quiesce DFSMSrmm, use one of the MVS MODIFY commands shown in Figure 7-25.

```
MODIFY DFRMM,QUIESCE
F DFRMM,QUIESCE
```

Figure 7-25 MVS MODIFY DFSMSrmm command to stop all tape activities

Important: If you have multiple z/OS images, you must repeat the MODIFY command for each z/OS image.

Example 7-8 shows a sample EDGBKUP JCL that can be used to reorganize your DFSMSrmm control data set after you see the message that the DFSMSrmm subsystem is quiesced.

Example 7-8 CDS reorganization using EDGBKUP

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='BACKUP(REORG)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9216
```

The following definitions apply to Example 7-8:

BACKUP DD Specifies to back up the control data set
MASTER DD Identifies the DFSMSrmm control data set
SYSPRINT DD Contains the utility program messages

Example 7-9 shows how to reorganize the control data set and back up the journal. DFSMSrmm backs up the journal before attempting to restore the control data set because the JRNLBKUP DD and the JOURNAL DD statement are specified in this example.

Example 7-9 CDS reorganization and backing up the journal using EDGBKUP

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='BACKUP(REORG)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//JOURNAL  DD   DISP=SHR,DSN=RMM.PROD.JRNL
//BACKUP   DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.BACKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9216
//JRNLBKUP DD   DISP=(,CATLG),UNIT=SYSDA,
//         SPACE=(CYL,(100,50),RLSE),
//         DSN=RMM.HSKP.JRNLBKUP(+1),
//         RECFM=VB,BLKSIZE=0,LRECL=9248
```

The following explanations apply to Example 7-9 on page 239:

BACKUP DD Specifies to back up the control data set

JOURNAL DD Identifies the DFSMSrmm journal to be used during backup processing

JRNLBKUP DD Specifies to back up the journal

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

After you check the success of the reorganization, restart the DFSMSrmm subsystem using an MVS MODIFY command as shown in Figure 7-26.

```
MODIFY DFRMM,M=xx
```

Figure 7-26 MVS MODIFY command to restart DFSMSrmm

Important: If you have multiple z/OS images, you must repeat the MODIFY command for each z/OS image. Do not use the /RO *ALL command to restart DFSMSrmm on multiple z/OS images at the same time to minimize the risk of enqueue conditions.

The following explanations apply to Figure 7-26:

DFRMM Is the name of the started task

M Specifies the parmlib member name that must be used

Note: To restart DFSMSrmm after it is quiesced, the operator must issue the command specified with a member name.

RESTORE

Specify RESTORE to restore the control data set from a backup copy. When you specify the BACKUP DD statement, DFSMSrmm restores the control data set. When you specify the JOURNAL DD statement, forward recovery is started. You can specify both the BACKUP DD statement and the JOURNAL DD statement in the same job step to restore and forward recover the control data set from journal backups and the journal data set.

The DFSMSrmm subsystem must be stopped or quiesced when you restore the active control data set:

- Restoring the control data set without forward recovery

  You can restore your DFSMSrmm control data set without forward recovery. If you do this, you can lose data if the backup copy is not an actual one, and no changes are made since the backup copy was created.

  Example 7-10 shows how you can use EDGBACK JCL to restore your DFSMSrmm control data set without forward recovery.

Example 7-10 Restoring a control data set without forward recovery

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(0)
```

The following explanations apply to Example 7-10 on page 240:

BACKUP DD Is the current actual backup copy of the control data set
MASTER DD Identifies the DFSMSrmm control data set
SYSPRINT DD Contains the utility program messages

Important: Use this example only if you have no need for a forward recovery.

- Restoring and forward recovery for the control data set

  You can restore your DFSMSrmm control data set and forward recover the DFSMSrmm control data set. This is the normal way to have the same information as before a crash.

  Example 7-11 shows how you can use EDGBACK JCL to restore your DFSMSrmm CDS and forward recover it from the journal data set.

Example 7-11 Restoring a control data set with forward recovery

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(0)
//JOURNAL  DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-11:

BACKUP DD Is the current actual backup copy of the control data set
JOURNAL DD Identifies the DFSMSrmm journal to be used for forward recovery
MASTER DD Identifies the DFSMSrmm control data set
SYSPRINT DD Contains the utility program messages

Example 7-12 shows how you can use EDGBACK JCL to restore your DFSMSrmm CDS and forward recover it using the backup copy minus one (-1) if your current DFSMSrmm CDS backup copy is destroyed or if it is unavailable.

Example 7-12 Restoring a control data set from backup copy -1 with forward recovery

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(-1)
//JOURNAL  DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(0)
//         DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-12:

BACKUP DD Is the current backup copy of the control data set

JOURNAL DD Identifies the newest DFSMSrmm journal backup copy and DFSMSrmm journal to be used for forward recovery

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

Important: If you forward recover using multiple journal data sets, concatenate the journal data sets in the order in which changes were originally made.

Example 7-13 shows how you can use EDGBACK JCL to restore your DFSMSrmm CDS from a backup copy -1 and forward recover it using the latest journal backup and the backup -1. This procedure can be used when the latest DFSMSrmm CDS backup copy is destroyed or not available.

Example 7-13 Restoring a control data set from backup copy - 2 with forward recovery

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(-2)
//JOURNAL  DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(-1)
//         DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(0)
//         DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-13:

BACKUP DD Specifies the backup copy - 2 of your DFSMSrmm control data set

JOURNAL DD Identifies the DFSMSrmm journal backup copy - 2, the newest DFSMSrmm journal backup copy, and the DFSMSrmm journal to be used for forward recovery

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

Important: If you forward recover using multiple journal data sets, concatenate the journal data sets in the order in which changes were originally made.

The situation is the same as in the previous example, but the backup is made using BACKUP(DSS), the temporary journal backup, and the later ones need to be applied in the correct order. Example 7-14 shows a sample JCL.

Example 7-14 Restoring a control data set from backup copy - 2 with forward recovery

```
//STEP0001 EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(-2)
//JOURNAL  DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(-1)
//         DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(0)
//         DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-14:

BACKUP DD Specifies the backup copy - 2 of your DFSMSrmm control data set

JOURNAL DD Specifies the backup copy - 2, backup copy -1, the newest backup copy of the DFSMSrmm journal, and the DFSMSrmm journal

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

Example 7-15 on page 243 shows the correct JCL to use if the DFSMSrmm CDS backup copy was created using EDGSKP(DSS) or EDGBKUP(DSS). In this case, the last backup copy of the journal and the actual journal must always be used for forward recovery.

Example 7-15 Restoring a control data set if BACKUP(DSS) was used

```
//EDGBKUP  EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//BACKUP   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(0)
//JOURNAL  DD   DISP=SHR,DSN=RMM.HSKP.JRNLBKUP(0)
//         DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-15:

BACKUP DD Is the current backup copy of the control data set

JOURNAL DD Identifies the newest DFSMSrmm journal backup copy and actual DFSMSrmm journal to be used for forward recovery

MASTER DD Identifies the DFSMSrmm control data set

SYSPRINT DD Contains the utility program messages

Restriction: When a backup is taken using BACKUP(DSS) and you restore from the latest control data set backup, both the latest journal backup and the active journal must be used for forward recovery.

If you did not perform forward recovery when you restored the CDS backup, you can perform the recovery with the restored CDS as shown in Example 7-16.

Example 7-16 Forward recovery of restored CDS

```
//EDGBKUP  EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//JOURNAL  DD   DISP=SHR,DSN=RMM.PROD.JRNL
```

The following explanations apply to Example 7-16:

JOURNAL DD Specifies the actual DFSMSrmm journal
MASTER DD Identifies the DFSMSrmm control data set
SYSPRINT DD Contains the utility program messages

7.4.3 EDGUTIL

Use the EDGUTIL utility to create or update the control data set control record. To create or update the control record, the user of EDGUTIL must have UPDATE or higher RACF access to the DFSMSrmm control data set.

After DFSMSrmm is implemented, you only need to use this utility to update the control record if you are implementing new functions, or if you must correct one of the counts stored in the control record. This section describes how you can use the utility to implement new functions.

CATSYNCH

The CATSYNCH function specifies whether the DFSMSrmm control data set and system catalogs are synchronized:

NO CATSYNCH(NO) clears the last synchronization date and time. Specify CATSYNCH(NO) to force synchronization of the DFSMSrmm control data set and user catalogs the next time that inventory management is run.

YES CATSYNCH(YES) sets the last synchronization date and time to the current date and time.

Example 7-17 shows a sample EDGUTIL JCL that you can use to set the last synchronization date and time to the current date and time.

Example 7-17 Set last synchronization date and time

```
//STEP0001 EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.PROD.CDS
//SYSIN    DD   *
 CONTROL CATSYNCH(YES)
/*
```

Note: Do not specify CATSYNCH(YES) if you have not run the EDGHSKP utility with the CATSYNCH parameter on each system to synchronize the DFSMSrmm control data set with the user catalogs.

EXTENDEDBIN

The EXTENDEDBIN function enables DFSMSrmm extended bin support, which allows the reuse of bins at the start of a move:

YES When extended bin support is enabled, DFSMSrmm records additional information in the volume and bin record while a volume is moving from or to a bin-managed storage location.

Note: Do not enable extended bin support until you have made sure that the same level of code has been installed for all the DFSMSrmm systems that share a control data set.

When it is enabled, extended bin support cannot be disabled.

STACKEDVOLUME

The STACKEDVOLUME function enables DFSMSrmm stacked volume support:

YES When you enable stacked volume support, DFSMSrmm uses the stacked volume records to manage the movement of the volumes that are contained in the stacked volumes.

Note: Before enabling stacked volume support, ensure that all systems using the control data set are at the same release level.

You cannot remove stacked volume support after it is enabled.

7.5 Initializing and erasing volumes

EDGINERS is the DFSMSrmm utility for initializing and erasing volumes. Use this utility as part of the daily inventory management tasks. You can run EDGINERS in manual or in automatic mode. The suggestion is to run in automatic mode unless you need to make any individual change to a volume.

You can use the LABEL procedure, which runs EDGINERS for initializing or erasing volumes manually. This procedure allows you to initialize and erase any tape volume whether it is defined to DFSMSrmm or not. If you initialize a volume that is not defined to DFSMSrmm, when it is initialized, it is automatically defined. The LABEL procedure is supplied as the EDGLABEL member in the SAMPLIB data set.

Avoid the manual use of EDGINERS by defining all the volumes to DFSMSrmm previously with the RMM ADDVOLUME command, and setting the pending release action to INIT. The following command adds 100 new scratch volumes starting on A00001. These volumes need to be labeled before they are used as the INIT(Y) operand indicates. The TSO RMM subcommand shown in Figure 7-27 can be used to add the volumes.

```
RMM AV A00001 COUNT(100) STATUS(SCRATCH) INIT(Y)
```

Figure 7-27 TSO ADDVOLUME subcommand to add 100 volumes

Or, if you have the volumes defined to DFSMSrmm, but not with the pending release action INIT, you can change it by using the command in Figure 7-28.

```
RMM CV A00001 INIT(Y)
```

Figure 7-28 TSO CHANGEVOLUME subcommand to set the INIT flag

You can also add duplicate volumes to DFSMSrmm to be initialized. However, you can also initialize duplicate volumes using EDGINERS, which automatically adds the volume to DFSMSrmm. Use the VOL1 parameter shown in Figure 7-29 if you add a duplicate volume.

```
RMM AV A00001 STATUS(SCRATCH) INIT(Y) VOL1(ABC001)
```

Figure 7-29 Adding a duplicate volume

EDGINERS automatic processing is enabled when the INIT and ERASE options are specified in the control data set. When the volumes have the pending release action INIT or ERASE, they can be processed automatically by DFSMSrmm. There are some required parameters in the EDGINERS JCL EXEC statement to be able to process any volume: COUNT, INITIALIZE, ERASE, MEDIANAME, POOL, or LOCATION. When EDGINERS is running as an automatic process, it does not require any operator intervention.

To initialize and erase volumes without operator intervention, you can set up automatic processing by specifying any of these EDGINERS EXEC parameters:

ALVER3 Use ALVER3 to set the EDGINERS processing default value for ISO/ANSI label tapes to Version 3.

ALVER4 Use ALVER4 to set the EDGINERS processing default value for ISO/ANSI label tapes to Version 4.

BATCH(*number_of_batches*)

Use BATCH to specify the number of batches of volumes to be processed in a single run of EDGINERS automatic processing. Use the COUNT parameter to specify the number of volumes in each batch. The COUNT is the number of volumes that are initialized or erased before DFSMSrmm verifies the volumes. After DFSMSrmm verifies the volumes in a batch, EDGINERS starts again to initialize or erase the volumes in the next batch.

The default for BATCH is BATCH(1). To process all volumes that have actions pending, specify BATCH(0).

COUNT(*count*)

Use COUNT to specify the number of volumes to initialize or to erase when DFSMSrmm performs automatic processing. Use COUNT with the BATCH parameter to specify the number of volumes in each batch of volumes to be processed. The maximum value that you can specify is 99.

The default value is 10.

ERASE(DSE|INIT|SHRED|SHREDDSE)

Use ERASE to request DFSMSrmm to select volumes that have the erase action pending. If automatic processing is in effect but ERASE is not specified, DFSMSrmm only selects volumes with the initialize action pending.

DSE

Specifies that a Data Secure Erase (DSE) needs to be attempted. This uses the tape drive hardware capability to erase data from the volume. This is the default value if the ERASE parameter is omitted or specified without a value.

INIT

Specifies that an ERASE action equates to an INIT action; no secure erase is attempted and the volume is relabeled as if the INIT action had been requested.

SHRED

For encrypted volumes, this value specifies that the Data Key must be made unusable by the drive. For non-encrypted volumes, the DSE action is attempted.

SHREDDSE

For encrypted volumes, this value specifies that the Data Key must be made unusable by the drive, and that any non-encrypted residual data on the volume will be subject to DSE. For non-encrypted volumes, the DSE action is attempted.

INITIALIZE

Use INITIALIZE to request DFSMSrmm to select volumes that have the initialize action pending. If automatic processing is in effect but neither INITIALIZE nor ERASE is specified, INITIALIZE is the default.

LOCATION(*library_name*)

Use LOCATION to specify a subset of volumes for automatic processing. The *library_name* must be the name of a system-managed tape library that is on the running system or SHELF. If you specify LOCATION, you cannot specify MEDIANAME, MEDIATYPE, POOL, or RECORDINGFORMAT.

MEDIANAME(medianame|parmlib_default_medianame)
Use MEDIANAME to specify a subset of volumes for automatic processing. If you specify MEDIANAME, you cannot specify LOCATION, MEDIATYPE, POOL, or RECORDINGFORMAT. If you do not specify MEDIANAME, MEDIATYPE, LOCATION, POOL, or RECORDINGFORMAT, DFSMSrmm uses MEDIANAME as the default parameter for automatic processing. This means that all volumes that are defined with the default medianame are selected if they have the required action pending.

MEDIATYPE(*media*) Use MEDIATYPE to specify a subset of volumes for automatic processing. Specifies the volume's physical media type. If you specify MEDIATYPE, you cannot specify LOCATION, MEDIANAME, POOL, or RECORDINGFORMAT. Use one of these types:

*****	The volume is not a cartridge.
CST	Cartridge System Tape.
ECCST	Enhanced Capacity Cartridge System Tape.
EHPCT	Extended High Performance Cartridge Tape.
HPCT	High Performance Cartridge Tape.
MEDIA5/ETC	IBM TotalStorage Enterprise Tape Cartridge.
MEDIA6/EWTC	IBM TotalStorage Enterprise WORM Tape Cartridge 3592.
MEDIA7/EETC	IBM TotalStorage Enterprise Economy Tape Cartridge 3592.
MEDIA8/EEWTC	IBM TotalStorage Enterprise Economy WORM Tape Cartridge 3592.
MEDIA9/EXTC	IBM TotalStorage Enterprise Extended Tape Cartridge 3592.
MEDIA10/EXWTC	IBM TotalStorage Enterprise Extended WORM Tape Cartridge 3592.
MEDIA11/EATC	IBM Enterprise Advanced Tape Cartridge.
MEDIA12/EAWTC	IBM Enterprise Advanced WORM Tape Cartridge.
MEDIA13/EAETC	IBM Enterprise Advanced Economy Cartridge.

POOL(*pool_prefix*) Use POOL to specify a subset of volumes for automatic processing. A pool prefix is one-to-five alphanumeric, national, or special characters followed by an asterisk (*). The pool must be one that is defined to DFSMSrmm on the running system. If you specify POOL, you cannot specify LOCATION, MEDIANAME, MEDIATYPE, or RECORDINGFORMAT.

RECORDINGFORMAT(*format*)
Use RECORDINGFORMAT to specify a subset of volumes for automatic processing. RECORDINGFORMAT specifies the basic recording format for tape volumes.

*****	An asterisk indicates that the format is unknown or that the volume is not a tape volume.
18TRACK	Data has been written to the volume in 18-track format. A recording format of 18TRACK is valid with MEDIATYPE(CST) and MEDIATYPE(ECCST) only.
36TRACK	Data has been written to the volume in 36-track format. A recording format of 36TRACK is valid with MEDIATYPE(CST) and MEDIATYPE(ECCST) only.

128TRACK Data has been written to the volume in 128-track format. A recording format of 128TRACK is valid with MEDIATYPE(EHPCT) and MEDIATYPE(HPCT) only.

256TRACK Data has been written to the volume in 256-track format. A recording format of 256TRACK is valid with MEDIATYPE(EHPCT) and MEDIATYPE(HPCT) only.

384TRACK Data has been written to the volume in 384-track format. A recording format of 384TRACK is valid with MEDIATYPE(EHPCT) and MEDIATYPE(HPCT) only.

EFMT1 Data has been written to the volume in EFMT1 (enterprise format 1) recording format. A recording format of EFMT1 is valid with MEDIATYPE(MEDIA5), MEDIATYPE(MEDIA6), MEDIATYPE(MEDIA7), and MEDIATYPE(MEDIA8) only.

EFMT2 Data has been written to the volume in EFMT2 (enterprise format 2) recording format. A recording format of EFMT2 is valid with MEDIATYPE(MEDIA5), MEDIATYPE(MEDIA6), MEDIATYPE(MEDIA7), MEDIATYPE(MEDIA8), MEDIATYPE(MEDIA9), and MEDIATYPE(MEDIA10) only.

EEFMT2 Data has been written to the volume in EEFMT2 (enterprise encrypted format 2) recording format. A recording format of EEFMT2 is valid with MEDIATYPE(MEDIA5), MEDIATYPE(MEDIA6), MEDIATYPE(MEDIA7), MEDIATYPE(MEDIA8), MEDIATYPE(MEDIA9), and MEDIATYPE(MEDIA10) only.

EFMT3 Data has been written to the volume in EFMT3 (enterprise format 3) recording format. A recording format of EFMT3 is valid with MEDIATYPE(MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9, and MEDIA10) only.

EEFMT3 Data has been written to the volume in EEFMT3 (enterprise encrypted format 3) recording format. A recording format of EEFMT3 is valid with MEDIATYPE(MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9, and MEDIA10) only.

EFMT4 Data has been written to the volume in EFMT4 (enterprise format 4) recording format. A recording format of EFMT4 is valid with MEDIATYPE(MEDIA9, MEDIA10, MEDIA11, MEDIA12, and MEDIA13) only.

EEFMT4 Data has been written to the volume in EEFMT4 (enterprise encrypted format 4) recording format. A recording format of EEFMT4 is valid with MEDIATYPE(MEDIA9, MEDIA10, MEDIA11, MEDIA12, and MEDIA13) only.

STATUS Use STATUS to control the kind of tapes that you want DFSMSrmm to initialize or erase. The default for STATUS is NOTMASTER. Specifying STATUS requests automatic processing.

ALL EDGINERS processes all volumes that have the INITIALIZE or ERASE action pending.

NOTMASTER EDGINERS processes all volumes in SCRATCH, USER, INIT, ENTRY, or PENDING RELEASE status that have the INITIALIZE or ERASE action pending

NOTMASTER is the default.

SCRATCH EDGINERS processes volumes in SCRATCH, INIT, ENTRY, or PENDING RELEASE status that have the INITIALIZE or ERASE action pending.

VERIFY|NOVERIFY Use VERIFY to request that DFSMSrmm ask the operator to remount each volume that has been successfully erased or labeled. The volumes are requested in reverse order, and the volume labels read to ensure that no operator errors have occurred, for example, a mismatch between the internal label and the external label. For automatic processing, VERIFY is the default. For manual processing, NOVERIFY is the default.

WRONGLABEL Use WRONGLABEL to specify the processing that DFSMSrmm performs when a wrong volume is mounted. WRONGLABEL processing does not apply to no label (NL) tapes. For NL tapes, DFSMSrmm issues the WTOR EDG6628A to obtain the volume serial number or rack number for the volume that has been mounted. You can use WRONGLABEL when you are running EDGINERS in automatic mode and manual mode.

FAIL DFSMSrmm does not prompt the operator to accept a mounted volume that does not match the requested volume. The mount request is rejected, the volume is unmounted, and DFSMSrmm issues message EDG6661E or message EDG6662E.

IGNORE When the wrong volume is mounted, DFSMSrmm does not issue any operator prompt. DFSMSrmm issues message EDG6661E or EDG6662E to log the relabeling and processing proceeds. This is an extremely dangerous option. Use it with caution because any volume can be relabeled as long as the requested volume has the INIT action or is not defined to DFSMSrmm. Use of this option requires CONTROL access to RACF FACILITY class resource STGADMIN.EDG.INERS.WRONGLABEL.

PROMPT When an incorrect volume label is detected by EDGINERS for the mounted volume, the operator is always prompted to confirm the processing to be performed. DFSMSrmm issues message EDG6661E or message EDG662E, followed by message EDG6663D. Processing continues according to the response to message EDG6663D.

RMMPROMPT When the volume serial number of the mounted volume that is defined to DFSMSrmm does not match the volume serial number of the requested volume, DFSMSrmm issues message EDG6663D to prompt the operator to confirm processing. If the magnetic volume serial number of the tape is not known to DFSMSrmm, initialization continues as if the tape had no magnetic label. If the volume is known to DFSMSrmm, DFSMSrmm issues messages EDG6662E and EDG6663D to prompt the operator or issues message EDG6661E to log the relabeling.

Example 7-18 shows sample JCL to initialize the next 20 volumes that have a MEDIANAME of 3490 without a requirement to remount each volume that has been successfully labeled.

Example 7-18 Sample JCL for automatic processing

```
//AUTOINIT EXEC PGM=EDGINERS,
//         PARM='MEDIANAME(3490),NOVERIFY,COUNT(20)'
//SYSPRINT DD   SYSOUT=*
//TAPE     DD   UNIT=(3490,,DEFER)
```

Note: When you request automatic processing, DFSMSrmm does not process any SYSIN commands you specify.

For more information about EDGINERS, and a complete description of the parameters, see *DFSMSrmm Guide and Reference,* SC26-7404.

7.6 Disposition processing

DFSMSrmm disposition processing is optional and provides support for the following functions:

- Providing operators with information to assist them in tasks, such as moving a tape to a specific location
- Generating sticky labels
- Updating the location where a volume resides

DFSMSrmm disposition processing occurs at CLOSE and end-of-volume (EOV) for each volume. DFSMSrmm does not provide support for asynchronous processing in a separate job step. To implement this function, see the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

To produce labels with DFSMSrmm, you can use the default output. It is a WTO on route code 13 or the OUTPUT JCL statement in the DFSMSrmm started procedure to dynamically allocate a SYSOUT file for each label.

Figure 7-30 shows the default label format for a tape cartridge.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7.
dsname_____________________________________
userdata_____________________________________________________________

 jobname_     crdate____
              expdt_____
 den
```

Figure 7-30 Default label format for a tape cartridge

The JCL in Example 7-19 shows this function using the default output for the label.

Example 7-19 Sample JCL to create a sticky label

```
//STEP1    EXEC PGM=IEBGENER,REGION=4M
//SYSIN    DD   DUMMY
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DSN=YCJRES1.DSN9,DISP=SHR
//SYSUT2   DD   DSN=YCJRES1.TEST.DISP,DISP=(,KEEP),
```

```
//         SPACE=(CYL,(2,2)),VOL=SER=TST108,UNIT=3590
//DISPDD DD *
0B90.TST108.0001.THIS IS A TEST
/*
```

When you run this sample JCL, you see the message shown in Figure 7-31.

```
EDG4054I 0B90.TST108.0001.THIS IS A TEST
```

Figure 7-31 EDG4054I disposition message

Figure 7-32 shows the sticky label.

```
YCJRES1.TEST.DISP


 YCJRES1      2001/052
              2001/055
      IDRC 00121  027951 FB

 TST108 0001 SL        0B90
```

Figure 7-32 Example label format

The sticky label shown in Figure 7-32 contains the following information:

dsname	YCJRES1.TEST.DISP
jobname	YCJRES1
crdate	2001/052
expdt	2001/055
comp	IDRC
lrecl	00121
blksize	027951
recfm	FB
volser	TST108
seqn	0001
lab	SL
devc	0B90

7.7 Software products and foreign tapes

DFSMSrmm provides support for software products and foreign tapes. To manage software tapes, you can define the product to DFSMSrmm, and DFSMSrmm then tracks the volume as a product volume.

To take advantage of product volume management, you need to set up some DFSMSrmm parmlib options, define rack numbers, define products to DFSMSrmm, and define the volumes that are associated with these products.

7.7.1 Defining software products

To define software products and the volumes that are associated with them, you can use DFSMSrmm ISPF panels or TSO RMM subcommands. In the examples, we used TSO RMM subcommands.

You need to follow these steps:

1. If you want to have a separate range for product volumes, define a new VLPOOL in the EDGRMMxx PARMLIB member. The following statement shows a VLPOOL where the prefix for the product volumes is PP. See Figure 7-33.

```
VLPOOL DESC('software products') EXPDTCHECK(Y)       -
       MEDIANAME(CARTS) PREFIX(PP*) RACF(N) TYPE(R)
```

Figure 7-33 EDGRMMnn VLPOOL parmlib command

2. Restart DFSMSrmm with the new PARMLIB member.
3. Define some empty rack numbers in the pool as shown in Figure 7-34.

```
RMM ADDRACK PP0001 COUNT(100)
```

Figure 7-34 RMM ADDRACK subcommand

4. Define the products that you have or plan to have, using the TSO RMM subcommand that is shown in Figure 7-35.

```
RMM ADDPRODUCT 5695-DF1 NAME('DFSMS') OWNER(YCJRES1)     -
    LEVEL(V02R10M00) DESCRIPTION('DFSMS Version 2.10')
```

Figure 7-35 RMM ADDPRODUCT subcommand

5. Use the TSO RMM ADDVOLUME subcommand to add volumes. The default for LEVEL is V01R01M00. If your level is different, you must specify it. Figure 7-36 shows how you can assign volumes to a product.

```
RMM ADDVOLUME DZ11C0 POOL(PP*) MEDIANAME(CARTS)                  -
    FEATCD(1234) STATUS(MASTER) NUMBER(5695-DF1) LEVEL(V02R10M00)
```

Figure 7-36 Assigning a volume to a product

Important: If you use the DFSMSrmm panels to add the volumes, DFSMSrmm uses the following defaults:

- Status: MASTER
- Retention period: 90 days
- Release action: Return to owner

7.7.2 Foreign tapes

DFSMSrmm can manage tapes that do not follow your naming convention. These volumes usually are those that are sent to you from another site.

DFSMSrmm manages foreign tapes no differently than any other tapes. It records data set names, provides VRS retention and movement controls, and has expiration release options to help you identify how to remove the foreign tapes from your library. The only volumes that cannot be defined to DFSMSrmm are those that are duplicates of volumes already defined to DFSMSrmm.

The method that you use to define foreign tapes to DFSMSrmm is basically the same method that you use to define software product tapes.

The following steps take you through an example of how to define foreign tapes:

1. Decide on a range of rack numbers to be used for foreign tapes that enter your installation. Consider whether you want to have a separate range for each type of media. If you have both tape reels and tape cartridges to manage as foreign tapes, and want each to have its own range of rack numbers, you need multiple pool definitions.
2. Define a new VLPOOL in the DFSMSrmm parmlib member. For example, if the prefix for your foreign volumes is FR, you need to add a statement as shown in Figure 7-37.

```
VLPOOL DESC('foreign cartridges') EXPDTCHECK(Y) -
       MEDIANAME(CARTS) PREFIX(FR*) RACF(N) TYPE(R)
```

Figure 7-37 EDGRMMnn VLPOOL command to add a new pool for foreign volumes

3. Restart DFSMSrmm with the new parmlib member using the modify operator command as shown in Figure 7-38.

```
F DFRMM,M=00
```

Figure 7-38 Restart DFRMM

4. Define some empty rack numbers in the pool by using the TSO RMM ADDRACK subcommand shown in Figure 7-39. You can also use the ISPF dialog for the definition.

```
RMM ADDRACK FR0001 COUNT(100)
```

Figure 7-39 Adding 100 rack numbers

5. When a tape volume is brought into your installation, you can define it to DFSMSrmm by adding it to POOL(FR*) with the TSO RMM ADDVOLUME subcommand as shown in Figure 7-40. In this example, volume BB0001 is retained for 90 days and is then released. The release action, RETURN, prevents the volume from going to the scratch pool. You can use it to identify volumes that must be removed from the library, and either returned to their owner or destroyed.

```
RMM ADDVOLUME BB0001 POOL(FR*) MEDIANAME(CARTS) -
    STATUS(MASTER) RETPD(90) RELEASEACTION(RETURN)
```

Figure 7-40 Adding foreign volume BB0001

6. To identify volumes waiting for the RETURN release action, use the SEARCHVOLUME subcommand in Figure 7-41.

```
RMM SEARCHVOLUME VOL(*) OWNER(*) LIMIT(*) STATUS(RELEASE) -
    ACTION(RETURN)
```

Figure 7-41 Search for all volumes with a release action RETURN

7. Use the list that is generated to pull the volumes from the library and return them to their owner.

8. When the volume is removed from the library, confirm to DFSMSrmm that it has been removed. Use the command in Figure 7-42 to confirm the release action for volume BB0001.

```
RMM CHANGEVOLUME BB0001 CRLSE(RETURN)
```

Figure 7-42 Confirm the return action

When the RETURN action is confirmed, the volume is deleted from the CDS, and the rack it occupied is now empty.

7.7.3 Bypass DFSMSrmm and storage management subsystem processing

You have to bypass the storage management subsystem (SMS) and optionally DFSMSrmm if you get a foreign tape with a VOLSER that you are using in an IBM system-managed automated tape library (ATL) or Virtual Tape Server (VTS). The reason to bypass SMS is that, if SMS finds the VOLSER in the VOLCAT, it sends the allocation to the identified library and ignores the UNIT parameter you have specified in your JCL. You cannot mount a manual drive when a physical volume has the same volume serial number as an SMS-managed tape volume.

Example 7-20 shows sample JCL that can be used to bypass DFSMSrmm and SMS to copy the first file of a foreign volume with a duplicate volume serial number to your VTS.

Example 7-20 Sample JCL to bypass DFSMSrmm and SMS

```
//S1       EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=(OLD,KEEP),              <=== must be OLD
//         DSN=FOREIGN.TAPE.DATASET.FILE1,
//         UNIT=3490,VOL=SER=V00031,
//         EXPDT=98000,                       <=== bypass DFSMSrmm
//         STORCLAS=DUPT§SMS                  <=== bypass SMS
//SYSUT2   DD   DISP=(,CATLG),
//         DSN=SCHLUM.FOREIGN.TAPE.DATASET.FILE1,
//         UNIT=VTS,DCB=*.SYSUT1
//SYSIN    DD   DUMMY
```

Restriction: To bypass DFSMSrmm, the user must have read access to one of the resources defined in the RACF class FACILITY:

- STGADMIN.EDG.IGNORE.TAPE.*
- STGADMIN.EDG.IGNORE.TAPE.RMM.*

7.8 DFSMSrmm volume pools

A *pool* is a group of rack numbers or volumes that share a common prefix. In DFSMSrmm, there are two categories of pools: *rack* and *scratch*:

- A *rack pool* is shelf space that can be assigned to hold any volumes. Although you can add scratch volumes to these pools, you cannot normally use these volumes to satisfy non-specific mount requests. A rack pool cannot be used with the DFSMSrmm system-based scratch pooling. You can use rack pools for these functions:

- Volumes that are temporarily brought into the library but are returned to the owner after a period of time
- Customer, foreign tapes, and software product volumes
- Scratch volumes for use with either DFSMSrmm exit-selected scratch pooling, or pools based on storage group

- A *scratch pool* is shelf space that is assigned to hold volumes for use with the DFSMSrmm system-based scratch pooling. The volumes that are assigned to this shelf space can be used to satisfy scratch requests as long as the volumes are in scratch status. Once the volume has been written to, it becomes a volume with MASTER status, until the volume is returned to scratch status. The volume remains in the same DFSMSrmm system-based scratch pool because it occupies the same shelf space regardless of status.

DFSMSrmm supports the following basic types of pooling:

- Pools of shelf space that are based on rack number prefixes. You can create different pools with different characteristics, such as the media name and management criteria. You must use shelf space pools to store volumes that do not match your installation selected volume ranges.
- Pools of volumes that are based on the volume serial number prefix. These volumes do not have a rack number or the rack number matches the volume serial number. Each range identifies different characteristics, such as media name and management criteria.
- Scratch pools can be one or more pools of volumes. Scratch pools can be based on name, SMS storage group, prefix, or system.
- When you pool by storage group and use SMS automatic class selection (ACS) processing to assign a storage group to a tape data set, or the volume is in a system-managed manual tape library, DFSMSrmm ensures that a volume from the correct storage group is mounted. You can use a storage group for scratch pooling for system-managed manual tape libraries and non-system-managed tape.

7.8.1 Defining pools

You must define any pool you plan to use with DFSMSrmm in the PARMLIB member EDGRMMxx using the VLPOOL command.

Define at least one pool using the VLPOOL command in the DFSMSrmm parmlib member. Use the pools to assign a media name for shelf space and volumes based on a prefix. Also, define a default pool to ensure that volumes and racks that do not match a more specific VLPOOL are added with correct media name and pool name.

You can use the following operands in the VLPOOL command:

AUTOSCRATCH(YES|NO)
Use this operand to override the scratch processing release action for volumes in this pool.

DESCRIPTION('pool description')
Describes the pool. Use any description from 1 - 32 characters.

EXPDTCHECK(Y|N|O)
Indicates the way to automate the processing of unexpired tape data sets at the pool level:

N If you do not want DFSMSrmm to check or validate the expiration date of tape data sets. This is the suggested value when using DFSMSrmm-managed scratch pools.

Y If you want to ensure that all unexpired tape data sets are never overwritten.

O If you do not want DFSMSrmm to take any action, but you want to allow the operator or automation software to reply as necessary to any WTO (IEC507D). This is the default value.

MASTEROVERWRITE(ADD|LAST|MATCH|USER)
Specify the VLPOOL MASTEROVERWRITE operand to control how DFSMSrmm allows the overwriting of a volume.

MEDIANAME(*media_name*)
Indicates a name or media type for all volumes in this pool. Some samples are: CARTS, TAPE, LONG, SHORT, 3480, 3490, and 3592. Any name is valid because DFSMSrmm does not check whether it is defined to MVS.

NAME(*pool_name*) Specify any name to identify the pool in the operator messages and tape drive displays. You must code the RACK(999) operand of MNTMSG command for modifying the messages adding the pool name.

PREFIX(*nnnnn)** Specifies a generic shelf location used in operator messages, RMM TSO commands, and the ISPF dialog. DFSMSrmm uses the prefix value to assign a newly defined volume to a scratch pool.

RACF(Y|N) Specifies the type of RACF support you want DFSMSrmm to provide for the volumes in the pool. If you want to create RACF tape profiles, specify Y and ensure that OPTION TPRACF(A|P) is specified.

RELEASEACTION(NOTIFY)
Use this operand with the NOTIFY value to automatically set the NOTIFY release action for all volumes in this pool. If you have an email address for the owner of the volume, DFSMSrmm sends the owner notification that the volume is pending release.

SYSID(*system_name*)
This operand associates a scratch pool with a particular system. DFSMSrmm matches the value with the SYSID operand of the OPTION command. If the VLPOOL SYSID matches the OPTION SYSID, the following conditions are true:

A volume from this pool can be used to satisfy a scratch mount request on this system.

A volume from any other pool where its SYSID also matches the OPTION SYSID can be used to satisfy a scratch request on this system.

A volume from a pool that has no explicit VLPOOL SYSID can be used to satisfy a scratch request on a system if there are no other VLPOOL definitions that have a SYSID matching the OPTION SYSID.

The SYSID values are ignored when a storage group is selected by ACS processing and when a pool is set for a non-specific volume request by EDGUX100 processing. The system level pools can still be used as long as SMS ACS processing does not assign a storage group and the EDGUX100 exit does not set a specific pool to be used.

TYPE(S|R) Specifies the type of pool. Use R for rack pools and S for scratch pools.

Figure 7-43 shows some sample pool definitions.

```
VLPOOL RACF(N) TYPE(R) EXPDTCHECK(O) MEDIANAME(CARTS)      -
       AUTOSCRATCH(YES) MASTEROVERWRITE(USER)              -
       RELEASEACTION(NOTIFY)                               -
       DESCRIPTION('3590 POOL') PREFIX(TST*)
VLPOOL RACF(N) TYPE(S) EXPDTCHECK(Y) MEDIANAME(CARTS)      -
       AUTOSCRATCH(YES) MASTEROVERWRITE(ADD)               -
       DESCRIPTION('3490 POOL') PREFIX(HGT*)
VLPOOL RACF(N) TYPE(S) EXPDTCHECK(Y) MEDIANAME(3490)       -
       DESCRIPTION('3490 VTS SCRATCH POOL')                -
       PREFIX(V*)
VLPOOL RACF(N) TYPE(S) EXPDTCHECK(O) MEDIANAME(CARTS)      -
       DESCRIPTION('DEFAULT POOL') PREFIX(*)
VLPOOL RACF(N) TYPE(R) EXPDTCHECK(Y) MEDIANAME(CARTS)      -
       DESCRIPTION('SOFTWARE PRODUCTS') PREFIX(PP*)
```

Figure 7-43 Sample pool definitions

For more details about the pools, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

7.8.2 Scratch pooling

With OS/390 V2R10, DFSMSrmm extends the choices you have when implementing scratch pooling. You can use any combination of the following methods and can decide at the individual request level which method is used:

- Use the DFSMSrmm system-based pooling VLPOOL command with SYSID. This enforces pooling based on prefix pools.
- Use the DFSMSrmm EDGUX100 installation exit to select a pool prefix. You can use any of the information in the system to make a decision about the pool to be used.
- Use pre-ACS processing to obtain the DFSMSrmm system-based pool prefix, or the EDGUX100 exit-selected pool prefix in the MSPOOL read-only variable.
- Use SMS ACS processing to base scratch pooling on tape storage group names. This support is provided for non-system-managed tape and for system-managed tape libraries:
  - For system-managed tape libraries, an SG corresponds with a library.
  - For non-system-managed tape, an SG corresponds with a scratch pool.

DFSMSrmm validates the scratch mount in the following order:

- If it is a system-managed ATL, leave validation to the system.
- Storage group assigned for call to SMS ACS routines.
- Pool assigned by EDGUX100 exit.
- DFSMSrmm system-based pooling (VLPOOL).

7.9 Deleting non-existent volumes

When you convert to DFSMSrmm, you might have converted information about volumes that you no longer have. Some vendor tape management systems use databases that must be formatted for each range of volumes to be managed. You might have stopped using some of these ranges but never reformatted the database to remove the volumes.

Running under DFSMSrmm, you have the flexibility to add and delete volumes or other resources at any time. Use the knowledge you have about your existing tape volume ranges to check that you have defined to DFSMSrmm only the volumes and rack numbers that you want to manage.

You can use the DFSMSrmm commands to build lists of commands to delete resources you do not want. When you delete volumes, consider deleting the rack numbers that are left in the empty status.

For example, you know that volumes 100000 - 199999 no longer exist but that they were converted to DFSMSrmm. Use the commands shown in Figure 7-44 to delete the volumes and rack numbers.

```
RMM SEARCHVOLUME VOLUME(1*) OWNER(*) LIMIT(*) -
    NOLIST CLIST(‘ RMM DV ‘ , ‘   FORCE’ )
EXEC EXEC.RMM LIST
RMM DELETERACK 100000 COUNT(99999)
RMM DELETERACK 199999 COUNT(1)
```

Figure 7-44 Delete non-existing volumes and racks

For information about the commands that are used, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405. You can repeat these commands for each range of volumes and rack numbers you want to delete.

You can avoid the need to do cleanup by deleting the volume information during conversion. Use the EDGCNXIT user exit from EDGCNVT to check for ranges of volumes you want deleted, and tell EDGCNVT to skip the volumes and data set records within those ranges.

7.10 Repairing actions

In a system-managed tape library environment, there are three places to record the tape volume information:

- Library manager database (LM DB)

  The LM DB records information about all tape volumes in the tape library, including logical volumes in order to control the IBM tape library.

- Tape configuration database (TCDB)

  The TCDB is an ICF catalog that contains information about tape volumes, which are in the connected tape libraries. The TCDB is always required when you control IBM tape libraries with DFSMS, regardless of the existence of DFSMSrmm. The TCDB is not needed if you plan to use basic tape library support to control your tape libraries.

- DFSMSrmm control data set

  DFSMSrmm uses this data set to record all volumes, not only tape library volumes, but also non-tape library tape volumes. More information is recorded in the CDS than in the TCDB. The control data set also contains information about tape data sets.

All tape volume information in the LM DB, TCDB, and DFSMSrmm CDS must be consistent. DFSMSrmm provides the EDGUTIL utility for you to maintain the CDS. Also, you can use the ISMF AUDIT function to compare the TCDB with the LM DB. Figure 7-45 on page 259 shows the difference between the audit process before DFSMSrmm OS/390 Release 10 and with Release 10.

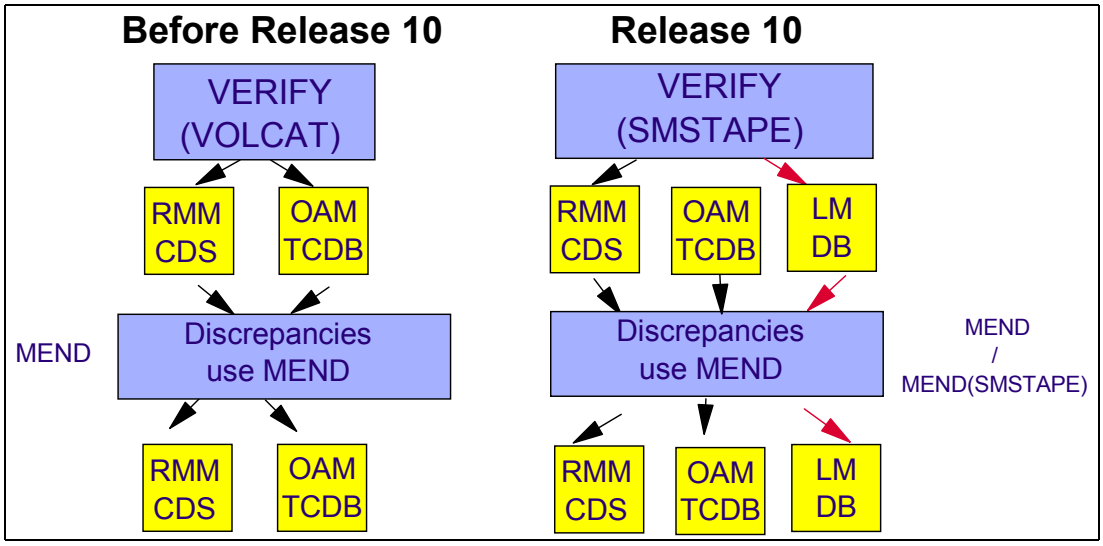


Figure 7-45 Three-way audit of CDS, TCDB, and LM DB

7.10.1 Using the VERIFY parameter

You can use the EDGUTIL utility with the VERIFY parameter to verify either active or inactive CDS. Figure 7-46 shows the VERIFY parameter syntax.

```
              |----------ALL----------|
              |   |----,----------|   |
              v   v               |   |
>>---VERIFY(-----------------------------------------------------------)---<<
                |-DSN------------|          |-,-|     |-VOLCAT---------|
                |-OWNER----------|                     |-SMS------------|
                |-PP-------------|
                |-RACK-----------|
                |-STORE----------|
                |-VOLUME---------|
                |-VRS------------|
```

Figure 7-46 EDGUTIL VERIFY syntax diagram

Verify processing can check the consistency between the records inside the DFSMSrmm CDS. In addition, the following actions occur:

- When you specify VERIFY(VOLCAT), DFSMSrmm checks the information between the CDS and the TCDB.
- When you specify VERIFY(SMSTAPE), DFSMSrmm checks the synchronization of the DFSMSrmm CDS with the TCDB, and in addition checks the LM DB, if needed.

  For VERIFY(SMSTAPE), DFSMSrmm scans both the DFSMSrmm control data set and the TCDB sequentially to ensure that volume records reside in both of them. If any system-managed volumes are only defined to DFSMSrmm CDS or the TCDB, DFSMSrmm checks it against the LM DB.

  DFSMSrmm also checks stacked volume information against the LM DB. Because the stacked volumes are not system-managed volumes, no information about stacked volumes resides in the TCDB. However, DFSMSrmm knows they need to be in the LM DB.

This function is available with the Release 10 enhancement, and it is meant to be the replacement of VERIFY(VOLCAT).

Example 7-21 shows a sample JCL to verify all the information of the CDS.

Example 7-21 JCL to verify all information in the CDS

```
//UTIL     EXEC PGM=EDGUTIL,PARM='VERIFY'
//SYSPRINT DD   DSN=RMM.YCJ4.MESSAGE.UTIL,DISP=SHR
//MASTER   DD   DSN=RMM.PROD.CDS,DISP=SHR
//SYSIN    DD   DUMMY
```

The default value for the VERIFY operand is ALL, but you can verify a specific type of record with the CDS, such as the data set information with operand DSN, and so on.

With the enhancement of DFSMSrmm OS/390 Release 10, you can perform a cross-check against DFSMSrmm, the TCDB, and the LM DB.

In z/OS V1.9, DFSMSrmm utility EDGUTIL is no longer considered to be part of inventory management processing. EDGUTIL can run at any time, even in parallel with other EDGUTIL instances. Also, a subset of volumes can be selected to be processed on a run of EDGUTIL. An optional SYSIN file allows you to select the subset from the available locations and volume entries during verification of volumes.

By default, all volumes are verified. The SYSIN commands in Figure 7-47 can be used to select the subset of volumes.

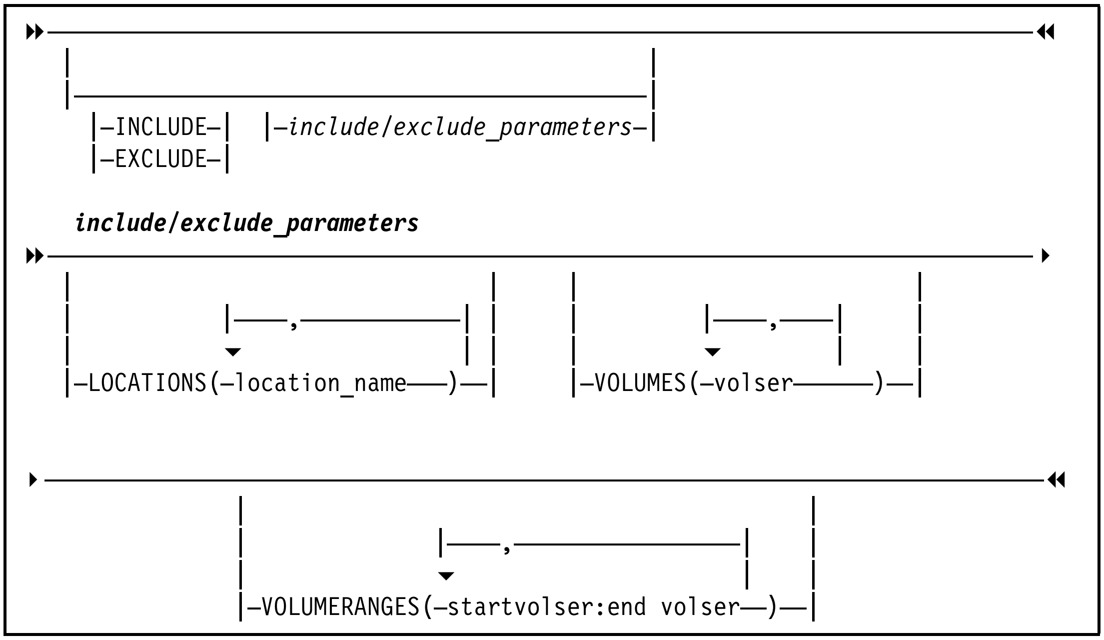


Figure 7-47 EDGUTIL SYSIN parameter syntax

7.10.2 Using the MEND parameter

You can only perform the MEND process with an inactive CDS.

When using the MEND parameter, the same verification as with VERIFY is performed and any error status found is corrected. Information in the LM DB or TCDB is not checked before the DFSMSrmm Release 10 enhancement.

If you successfully enabled stacked volume support with the EDGUTIL UPDATE option STACKEDVOLUME(YES), during MEND processing, DFSMSrmm creates stacked volumes in the DFSMSrmm control data set.

When you run MEND for the first time after enabling stacked volume support, DFSMSrmm needs MEDIANAME information for the stacked volumes it creates from the container information. MEDIANAME information is defined in the VLPOOL statements in parmlib. To ensure that this is available for EDGUTIL, you must have started DFSMSrmm at least once with the correct VLPOOL definitions before running EDGUTIL MEND.

Figure 7-48 shows the MEND parameter syntax.

Restriction: Do not enable stacked volume support until all systems that are using the control data set are on a supporting release level.

```
              |————————ALL————————|
              |   |———,————————|   |
              ▼   ▼              |   |
▶▶———MEND(————————————————————————————————————————————————————)———◀◀
                |—DSN——————————|         |—,—|    |—VOLCAT————————|
                |—OWNER————————|                    |—SMS——————————|
                |—PP———————————|
                |—RACK—————————|
                |—STORE————————|
                |—VOLUME———————|
                |—VRS——————————|
```

Figure 7-48 EDGUTIL MEND syntax diagram

The default value for the MEND operand is ALL; however, you can mend a subset of specific record types to increase the performance. Example 7-22 shows a sample JCL to MEND data set and volume records.

Example 7-22 JCL to MEND data set and volume records only

```
//UTIL     EXEC PGM=EDGUTIL,PARM='MEND(VOLUME,DSN)'
//SYSPRINT DD   DSN=RMM.YCJ4.MESSAGE.UTIL,DISP=SHR
//MASTER   DD   DSN=RMM.PROD.CDS,DISP=SHR
```

After MEND processing is completed, you might need to add or change some volume information. Use the RMM CHANGEVOLUME subcommand to set or change any information.

With the DFSMSrmm Release 10 enhancement, MEND processing can also correct CDS information from the TCDB or LM DB. Optionally, it can create a stacked volume record in order to support the VTS enhanced support and perform a three-way audit to check the information about stacked volumes and logical volumes (see Figure 7-49 on page 262).

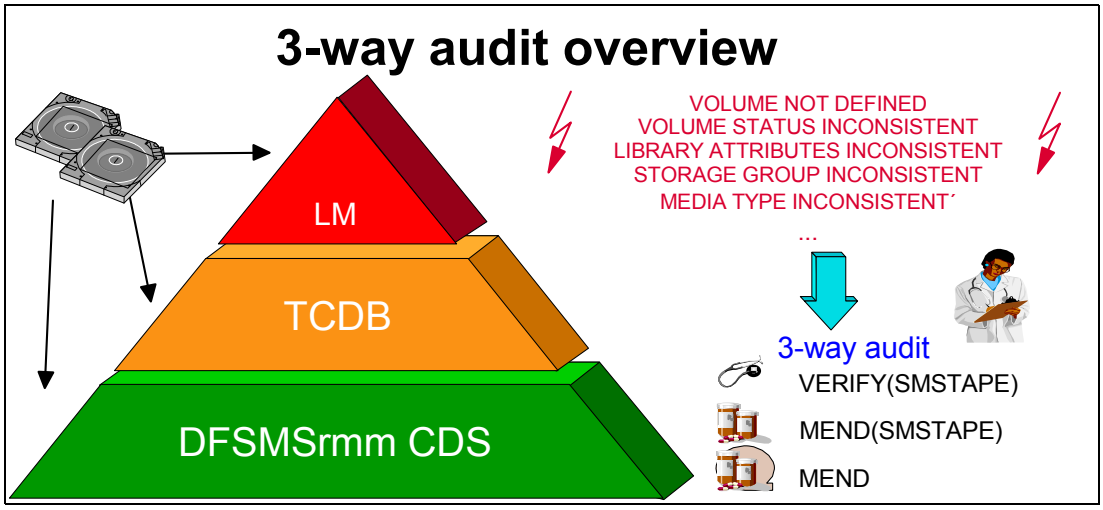


Figure 7-49 DFSMSrmm CDS, Volume Catalog, and Library Manager consistency check

When you specify MEND(SMSTAPE), DFSMSrmm performs the same verification action as VERIFY(SMSTAPE) and corrects any error status that is found. This processing uses information from the DFSMSrmm CDS for volume status and storage groups to correct the TCDB and LM DB. Figure 7-50 compares the MEND and MEND(SMSTAPE) processing.

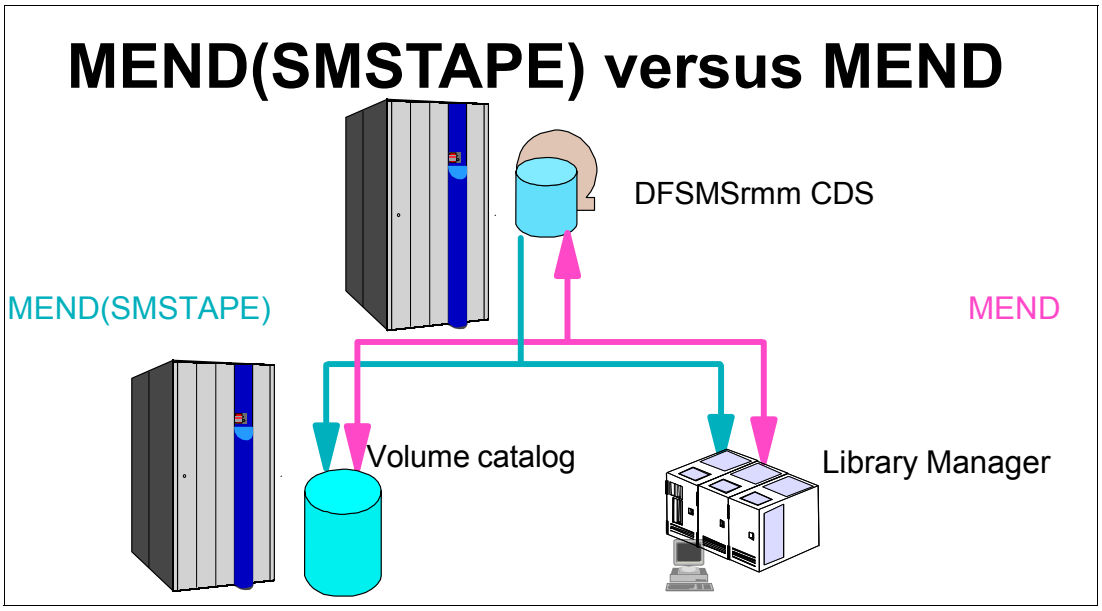


Figure 7-50 EDGUTIL MEND(SMSTAPE) does not update the CDS

You can run MEND(SMSTAPE) processing against either an active or inactive CDS. This function is available with the DFSMSrmm Release 10 enhancement.

In z/OS V1.9, the DFSMSrmm utility EDGUTIL is no longer considered to be part of inventory management processing. EDGUTIL can run at any time, even in parallel with other EDGUTIL instances. A subset of volumes can also be selected to be processed on a run of EDGUTIL. An optional SYSIN file allows you to select the subset from the available locations and volume entries during the verification of volumes.

By default, all volumes are verified. The SYSIN commands in Figure 7-51 on page 263 can be used to select the subset of volumes.

```
▶▶──────────────────────────────────────────────────────────────────────────────◀◀
  │                                                │
  │────────────────────────────────────────────────│
    │─INCLUDE─│  │─include/exclude_parameters─│
    │─EXCLUDE─│

   include/exclude_parameters
▶▶──────────────────────────────────────────────────────────────────────────────▶
  │                                │     │                                │
  │          │────,────────────│ │     │         │────,────│     │
  │          ▼                   │ │     │         ▼        │     │
  │─LOCATIONS(─location_name───)──│     │─VOLUMES(─volser──────)──│


▶──────────────────────────────────────────────────────────────────────────────◀◀
              │                                        │
              │             │────,─────────────────│    │
              │             ▼                      │    │
              │─VOLUMERANGES(─startvolser:end volser──)──│
```

Figure 7-51 EDGUTIL SYSIN parameter syntax

Deferred correcting TCDB and LM database

Correction of the tape configuration database (TCDB) and the library management database (LM DB) based on the RMM CDS information was previously done by using EDGUTIL with PARM=‘MEND(SMSTAPE)’. The correction was done synchronously with the verify processing, resulting in a long elapsed time.

In z/OS V1R9, by using EDGUTIL with PARM=‘VERIFY(SMSTAPE)’ and EDGSPLCS DD and the EDGSPLCS utility, the following actions occur:

- Control statements are generated in the EDGSPLCS output file by EDGUTIL.
- Volumes can be corrected after EDGUTIL processing is completed by using these statements as an input for the EDGSPLCS utility.

Figure 7-52 is an example of using the deferred TCDB and LM database correction method.

```
//HSKP     EXEC PGM=EDGUTIL,PARM='VERIFY(SMSTAPE)'
//EDGSPLCS DD   DISP=SHR,DSN=NAIDOO.SPLCS.DATA
//SYSPRINT DD   SYSOUT=*
//SYSIN DD *
INCLUDE VOLUMES(M*,V*,T*)
/*
```

Figure 7-52 EDGUTIL VERIFY(SMSTAPE) for a specific number of tapes

Note: The EDGSPLCS DD is written during VERIFY(SMSTAPE). It causes EDGUTIL to create statements to be used with the EDGSPLCS utility, if specified.

Figure 7-53 on page 264 is the output from SYSPRINT DD.

```
INCLUDE VOLUMES(M*,V*,T*)
EDG6433I STARTING VERIFICATION OF VOLUME   RECORDS
EDG6824I VOLUME TST020 IS IN VOLUME CATALOG ERROR STATUS 0004 SECURITY CONFLICT
EDG6846I VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR
VOLUME TST020
EDG6824I VOLUME TST021 IS IN VOLUME CATALOG ERROR STATUS 0004 SECURITY CONFLICT
EDG6846I VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR
VOLUME TST021
EDG6418W CONTROL DATA SET VERIFY COMPLETED WITH ERRORS
EDG6901I UTILITY EDGUTIL COMPLETED WITH RETURN CODE 4
```

Figure 7-53 SYSPRINT DD output

The following explanations apply to Figure 7-53:

EDG6846I `VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR VOLUME volser`

Explanation You are running the DFSMSrmm EDGUTIL utility to VERIFY SMS information for volumes. DFSMSrmm found inconsistencies between the DFSMSrmm information for the volume and the volume catalog and Library Manager database. This message indicates that the error can be corrected. A control statement has been written to the data set represented by the EDGSPLCS DD statement.

In the message text, the following information is shown:

volser This is the volume serial number of the volume to be corrected.

System action DFSMSrmm utility continues. A control statement has been written to the data set specified by the EDGSPLCS DD statement.

Operator response None.

Application Programmer Response Use the EDGSPLCS utility to process the statements produced. You can review the contents of the EDGSPLCS file and if you want to make those corrections, pass the file to the EDGSPLCS INDD file.

Source DFSMSrmm

Detecting Module EDGUTIL

Figure 7-54 shows the corresponding EDGSPLCS generated statements.

```
S  TST020              LIB1
P  TST021              LIB1
```

Figure 7-54 EDGSPLCS DD statements

7.10.3 Using EDGSPLCS

You can use the EDGSPLCS utility to issue supported commands to the Object Access Method (OAM) for system-managed volumes. DFSMSrmm builds the input commands for this utility automatically during EDGUTIL VERIFY(SMSTAPE) processing and EDGHSKP EXPROC processing when you request them.

You can run multiple copies of EDGSPLCS. Using different parameters, EDGSPLCS can be processing in parallel for multiple libraries, but this utility does not ensure that each parameter is different from any other currently running.

You need RACF ALTER authority to the relevant volume catalog in order to use the EDGSPLCS utility to update the TCDB. For example, if you use just one volume catalog and use the default volume catalog prefix, you need ALTER access to SYS1.VOLCAT.VGENERAL.

Running EDGSPLCS

Example 7-1 shows the sample JCL for the execution of EDGSPLCS. The INDD file in the JCL is partly created from the running programs described in “Deferred correcting TCDB and LM database” on page 263.

```
//EDGSPLCS EXEC PGM=EDGSPLCS,PARM='ACTION(S,P)'
//OUTDD    DD   SYSOUT=*
//INDD     DD   DISP=SHR,DSN=NAIDOO.SPLCS.DATA
```

Figure 7-55 EDGSPLCS JCL

You can also specify INDD DD * to specify some control cards directly as shown in Figure 7-56.

Restriction: The position of the action characters and the operands for INDD are fixed and cannot be changed.

```
//EDGSPLCS EXEC PGM=EDGSPLCS,PARM='ACTION(S,P)'
//OUTDD    DD   SYSOUT=*
//INDD     DD   *
P  THS000                LIB2
S  THS000                LIB2
/*
```

*Figure 7-56 EDGSPLCS JCL example with DD **

EXEC parameters for EDGSPLCS

Example 7-23 shows the execution parameters for EDGSPLCS.

Example 7-23 EDGSPLCS EXEC parameters

```
     |----ACTION(ALL)-------------|
>>-------------------------------------------------------------------------<<
     |----------------------------|       |----LOCATION(-library_name-)-|
         |                        |
         |          |----,--|     |
         |          v      |      |
         |--ACTION(--S-------)---|
                    |-P-|
                    |-I-|
                    |-X-|
                    |-M-|
                    |-E-|
```

The following explanations apply to Example 7-23 on page 265:

ACTION(ALL/S/P/I/X/M/E)

Specifies that only the specified requests in the input file are processed. You can optionally provide the name of a library to restrict the processing to only those requests. You can specify one or more of the possible actions:

- **S** Set to Scratch status
- **P** Set to Private status
- **I** Import volume
- **X** Export volume
- **M** Manual cartridge entry
- **E** Eject volume

If you do not specify the ACTION parameter, the default value is ALL.

LOCATION(*library_name*)

Specifies the name of the system-managed library for which the EDGSPLCS utility will process commands during this run. By default, all locations are considered. However, you can select a subset based on the library name by using this parameter.

INDD input file

Figure 7-57 displays an existing file of LRECL 80 containing control statements that direct the processing that the EDGSPLCS utility performs. The control statements in INDD have the format that is shown in Table 7-3.

```
av volser options        library   message
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
12 4      1               2        3
          1               7        6

S  TST123                 ATLLIB1
P  TST221                 VTS2
```

Figure 7-57 EDGSPLCS input record format

Table 7-3 shows all control statements and operands that can be used in the INDD input file for the EDGSPLCS utility.

Table 7-3 INDD control statements

Symbol	Explanation	Values
a	Action character	One of these values: **S** Set to Scratch status **P** Set to Private status **I** Import volume **X** Export volume **M** Manual Cartridge Entry **E** Eject volume
v	Verify request	One of these values: **V** Verify volume is resident blank Do not verify volume is resident
volser	Volume serial	

Symbol	Explanation	Values
options	Depends on action character	Action-specific values starting in column 11: **S** **P** Storage group name or blank followed by optional owner ID or blank. 'Owner ID' starts in column 19. **I** Cancel request. Specify C to cancel an existing import. Distributed library name to start an import in a specific library of a Peer-to-Peer (PtP) VTS. **X** Cancel request. Specify C to cancel an existing export. Distributed library name to start an export in a specific library of a PtP VTS. **M** Library name into which the volume is to be entered. Column 20; media type of volume to be entered, for example, 5. **E** Eject destination. Either C (convenience) or B (high capacity).
library	Library name	Starting in column 27, this is an eight-character field for you to specify the library name. This field is used by the LOCATION execution parameter and is only required if the LOCATION parameter is specified.
message	Output area for EDGSPLCS	After processing, this area contains a function-specific message from the EDGSPLCS utility.

OUTDD output file

This is the output file that is written by the EDGSPLCS utility. It contains a copy of each of the input control statements, and each statement contains a completion message as shown in Figure 7-58. The return and reason codes the EDGSPLCS utility has added (starting in column 36) are described in Table 7-4 on page 268.

```
av volser options        library   message
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
12 4      1               2        3
          1               7        6

SV THS012                 LIB2     QVR   RETURN CODE = 0004, REASON CODE = 0063.
S  THS013                          CUA   RETURN CODE = 0012, REASON CODE = 0063.
S  THS015                 LIB2     USE ATTRIBUTE NOT CHANGED
```

Figure 7-58 EDGSPLCS output record format

For the descriptions of the return and reason codes, see Chapter 6. "Library Control System (LCS) External Services", in the *DFSMS OAM Planning, Installation, and Storage Administration Guide for Tape Libraries,* SC35-0426.

Return codes for EDGSPLCS

EDGSPLCS issues the return codes shown in Table 7-4 on page 268.

Table 7-4 EDGSPLCS return codes

Return code	Explanation
0	All requested functions completed successfully. See the OUTDD records for individual messages from the input actions.
4	DFSMSrmm encountered a minor error during processing. See the OUTDD records for individual messages from the input actions.
8	At least one requested action was not supported. See the OUTDD records for individual messages from the input actions.
12	DFSMSrmm encountered a severe error during processing. DFSMSrmm stops the utility.

Reason codes for EDGSPLCS

EDGSPLCS issues the reason codes shown in Table 7-5.

Table 7-5 EDGSPLCS reason codes

Reason code	Explanation
0	SUCCESSFUL EXECUTION
2	OAM CONTROL BLOCKS NOT AVAILABLE
3	DELETED WITH MTL SOFTWARE-ONLY
4	VOLUME ALREADY SCRATCH
5	VOLUME ALREADY PRIVATE
6	CUA PROCESSING DISABLED
7	CUA PROCESSING NOT PERFORMED FOR THIS VOLUME PER INSTALLATION EXIT REQUEST
8	SCRATCH VOLUME THRESHOLD MESSAGE PROCESSING WAS NOT COMPLETED SUCCESSFULLY
9	REQUIRED TYPE PARAMETER NOT SPECIFIED
10	MUTUALLY EXCLUSIVE REQUIRED PARAMETERS SPECIFIED
11	INVALID TYPE VALUE SPECIFIED
12	REQUIRED FUNC PARAMETER NOT SPECIFIED
13	INVALID FUNC VALUE
14	REQUIRED USE PARAMETER NOT SPECIFIED
15	INVALID USE VALUE
16	REQUIRED VOLUME PARAMETER NOT SPECIFIED
17	INVALID VOLUME VALUE
18	REQUIRED UCBPTR NOT SPECIFIED
19	INVALID UCBPTR VALUE SPECIFIED
20	REQUIRED VOLLIST NOT SPECIFIED
21	INVALID VOLUME LIST VALUE

Reason code	Explanation
22	INVALID VALUE IN STORAGE GROUP LIST HEADER
23	REQUIRED LIBRARY NAME NOT SPECIFIED
24	INVALID VOLUME LIST, MIXED MEDIA, REWRITABLE AND WORM VOLUMES
25	INVALID EXPIRATION DATE VALUE
26	LIBRARY NOT DEFINED TO STORAGE GROUP
27	INVALID LIBRARY NAME SPECIFIED
28	INVALID WRITE PROTECT STATUS VALUE
29	INVALID ADDRESS SPECIFIED FOR PARAMETER LIST OR MAPPING MACRO
30	ADDRESS NOT ON WORD BOUNDARY OR LEVEL
31	REQUIRED TAPE DEVICE SELECTION INFORMATION (TDSI) NOT SPECIFIED
32	REQUIRED LIBRARY ID OR LIBRARY NAME NOT SPECIFIED
33	INVALID VALUE SPECIFIED FOR LIBRARY ID
34	INVALID STORAGE GROUP NAME
35	REQUIRED MEDIA TYPE NOT SPECIFIED FOR MCE VOLUME
38	INVALID COMPACTION SPECIFIED IN TDSI
39	INVALID SPECIAL ATTRIBUTE SPECIFIED IN TDSI
40	INVALID COMBINATION OF TAPE DEVICE SELECTION VALUES SPECIFIED
41	AMBIGUOUS TDSI COMBINATION SPECIFIED
42	TAPE DEVICE SELECTION VALUE SPECIFIED WHERE NOT ALLOWED OR NOT APPLICABLE
43	INVALID POINTER TO TDSI SPECIFIED
44	INVALID VALUE SPECIFIED FOR DISP KEYWORD
45	LIBRARY NAME AS DEFINED IN VOLUME RECORD NOT FOUND IN TCDB
46	NO ENABLED STORAGE GROUPS
47	NOT ALL VOLUMES ASSOCIATED WITH THE SAME STORAGE GROUP
48	STORAGE GROUP STATE IS NOTCON, DISALL, OR DISNEW
49	NO DEVICE POOLS EXIST TO FULFILL REQUEST FOR TDSI SPECIFICATION
51	SPECIFIC VOLSER REQUEST FOR SCRATCH VOLUME
52	VOLUMES RESIDE OUTSIDE LIBRARY
53	LIBRARY FOR SPECIFIED VOLUME NOT DEFINED TO SMS CONFIGURATION
54	SMS STORAGE GROUP WAS NOT OF TYPE TAPE
55	REQUESTED DEVICE DOES NOT RESIDE IN SAME LIBRARY AS REQUESTED VOLUME
56	NO LIBRARIES ASSOCIATED WITH LIST OF STORAGE GROUPS OR LIBRARY IS UNKNOWN

Reason code	Explanation
58	FAILURE ACCESSING THE VOLUME RECORD IN THE CATALOG
59	FAILURE ACCESSING THE LIBRARY RECORD IN THE CATALOG
60	FAILURE ACCESSING THE SMS STORAGE GROUP CONSTRUCTS
61	FAILURE ACCESSING HARDWARE CONFIGURATION INFORMATION
62	SPECIFIED LIBRARY IS NOT DEFINED TO ACTIVE SMS CONFIGURATION
63	VOLUME RECORD NOT FOUND FOR REQUESTED VOLUME
64	LIBRARY RECORD NOT FOUND IN TCDB FOR REQUESTED LIBRARY
65	LIBRARY LOGICAL TYPE NOT DEFINED
66	NO DEVICE POOLS TO FULFILL REQUEST FOR SPECIFIED RECORDING TECHNOLOGY
67	NO DEVICE POOLS TO FULFILL REQUEST FOR SPECIFIED MEDIA TYPE
69	REQUEST FAILED BECAUSE VOLUME NOT IN LIBRARY INSTALLATION EXIT
70	VOLUME NOT FOUND IN LIBRARY MANAGER INVENTORY
72	DELETED WITH MTL SOFTWARE-ONLY
74	REQUEST FAILED BECAUSE VOLUME SERIAL NUMBER ALREADY EXISTS IN LIBRARY MANAGER INVENTORY
75	UNEXPECTED UCBSCAN ERROR ENCOUNTERED DURING PROCESSING
76	DELETED WITH MTL SOFTWARE-ONLY
77	DELETED WITH MTL SOFTWARE-ONLY
78	ERROR ATTEMPTING TO RETRIEVE VOLUME RECORD
79	ERROR ATTEMPTING TO WRITE VOLUME RECORD
80	ESTAE ROUTINE NOT ESTABLISHED
81	GETMAIN FAILED FOR DEVICE POOL NAMES LIST OR LOCAL WORKING STORAGE
82	ABNORMAL TERMINATION OCCURRED DURING INSTALLATION EXIT (CBRUXCUA) EXECUTION
83	INVALID RETURN CODE OR DATA RETURNED FROM CHANGE USE ATTRIBUTE INSTALLATION EXIT (CBRUXCUA)
84	ABNORMAL TERMINATION OCCURRED DURING EXECUTION
90	CARTRIDGE ENTRY PROCESSING HAS BEEN DISABLED
91	CARTRIDGE ENTRY PROCESSING HAS BEEN SUSPENDED FOLLOWING ERROR INVOKING INSTALLATION EXIT
92	LIBRARY FOR MCE NOT MANUAL TAPE LIBRARY
93	LIBRARY OFFLINE, PENDING OFFLINE, OR NOT OPERATIONAL
94	SPECIFIED VOLUME ALREADY RESIDES IN ANOTHER LIBRARY
95	INSTALLATION EXIT VETOED ENTRY OF VOLUME INTO LIBRARY

Reason code	Explanation
96	VOLUME NOT ENTERED INTO MTL BECAUSE INSTALLATION EXIT SAID TO IGNORE THE VOLUME
97	VOLUME OF SAME VOLSER IS KNOWN DASD VOLUME
120	VOLUME IS INELIGIBLE BECAUSE THE TYPE OF MEDIA DEFINED IN THE VOLUME RECORD MAY NOT BE MOUNTED ON SPECIFIED DEVICE
121	VOLUME IS INELIGIBLE BECAUSE THE TYPE OF MEDIA DEFINED IN THE TDSI DOES NOT MATCH MEDIA DEFINED ON THE VOLUME RECORD
122	VOLUME IS INELIGIBLE BECAUSE THE VOLUME RECORD REFLECTS AN ERROR STATUS
123	VOLUME IS INELIGIBLE BECAUSE THE SPECIFIED RECORDING TECHNOLOGY IS INCOMPATIBLE WITH THE VOLUME MEDIA TYPE OR THE SPECIFIED DRIVE TYPE
130	SPECIFIED VOLUME ALREADY RESIDES IN THIS TAPE LIBRARY
131	SCRATCH VOLUME THRESHOLD PROCESSING NOT PERFORMED BECAUSE LIBRARY WAS NOT OPERATIONAL
132	DELETED WITH 3590 SUPPORT
133	DELETED WITH 3590 SUPPORT
134	NO TCDB RECORD BUT VOLUME RESIDES IN SPECIFIED LIBRARY
135	LIBRARY MISMATCH, VOLUME RESIDES IN SPECIFIED LIBRARY
136	LIBRARY MISMATCH, VOLUME NOT FOUND IN SPECIFIED LIBRARY
137	LIBRARY MISMATCH, UNABLE TO ACCESS SPECIFIED LIBRARY
138	LIBRARY SCRATCH COUNT NOT UPDATED IN TCDB
139	LIBRARY NAME MISMATCH, SPECIFIED LIB NAME DID NOT MATCH MTL VOLUME RECORD LIB NAME
201	VOLUME ALREADY EJECTED
202	INVALID VALUE FOR EJECT OPTION
203	INVALID VALUE FOR BULK EJECT
204	INVALID TSO USERID SPECIFIED
215	NOT ALL VOLUMES HAVE THE SAME RECORDING TECHNOLOGY
216	INVALID TCDBCHK=NO SPECIFIED WITH MTL LIB NAME
217	AT LEAST ONE OPTIONAL KEYWORD MUST BE SPECIFIED WITH FUNCTION
218	REQUIRED DATATYPE NOT SPECIFIED
219	INVALID POLICY NAME
220	MUTUALLY EXCLUSIVE OPTIONAL KEYWORDS SPECIFIED
221	INVALID STORAGE HEADER AND/OR LENGTH
300	OAM ABEND DURING EJECT REQUEST
302	EJECT REQUEST ALREADY PENDING FOR VOLUME

Reason code	Explanation
303	UNABLE TO MAKE USER ADDRESS SPACE NON-SWAPPABLE
304	TCDB ACCESS ERROR IN OAM
305	TCDB AUTHORIZATION ERROR IN OAM
306	OAM INTERNAL ERROR
307	VOLSER NOT IN TCDB
310	MEDIA TYPE OR RECORDING TECHNOLOGY NOT SUPPORTED AT THIS SOFTWARE LEVEL
311	DELETED WITH MTL SOFTWARE ONLY
312	FUNCTION NOT COMPATIBLE WITH THE LIBRARY
313	VOLUME IS CURRENTLY IN USE
314	IMPORT/EXPORT ALREADY IN PROGRESS OR HOST PROCESSING NOT COMPLETE
315	NOT ENOUGH PHYSICAL DRIVES AVAILABLE IN VTS
316	IMPORT/EXPORT NOT IN PROGRESS
317	EMPTY CATEGORY IMPORT: NO IMPORT VOLUMES EXPORT/IMPORT: NO SCRATCH VOLUMES
318	MAXIMUM LOGICALS DEFINED TO LIBRARY
319	NO TCDB RECORD AND VOLUME NOT FOUND IN SPECIFIED LIBRARY
320	NO TCDB RECORD AND UNABLE TO ACCESS SPECIFIED LIBRARY
321	MANUAL CARTRIDGE ENTRY FAILED MEDIA TYPE RETURNED FROM THE INSTALLATION EXIT DOES NOT MATCH THE MEDIA TYPE DEFINED IN THE VOLUME RECORD.
322	COMMAND REJECTED BY THE LIBRARY
323	UNABLE TO RETRIEVE POLICY NAMES FROM LIBRARY
324	VOLUME EXPIRE TIME HAS NOT ELAPSED
325	I/O TERMINATED DUE TO TIMEOUT DETECTION
400	OAM INITIALIZED WITH NULL CONFIGURATION
401	LIBRARY NOT ACCESSIBLE, OFFLINE, OR NOT OPERATIONAL
402	VISION SYSTEM NOT OPERATIONAL
403	EJECT PROCESSING HAS BEEN DISABLED BECAUSE AN ERROR IN THE EJECT INSTALLATION EXIT (CBRUXEJC) HAS BEEN DETECTED
404	OAM ADDRESS SPACE NOT AVAILABLE

7.10.4 DFSMSrmm CDS and DFSMShsm consistency checking

The DFSMSrmm CDS is the most important part of your DFSMSrmm environment. Ensuring availability of your DFSMSrmm controlled data requires ensuring the health of your control data set. This assurance can be achieved by running the standard DFSMSrmm EDGUTIL VERIFY against the DFSMSrmm CDS. You can also run Mainstar:*Fast*Audit/390 Tape Audit facility, a selectable unit of the Mainstar:*Fast*Audit/390 product suite.

The Mainstar *Fast*Audit/390 product is widely known and used in the z/OS and OS/390 environment. The Tape Audit function was developed to ensure that your DFSMSrmm CDS is healthy, in good shape, and that all data controlled by DFSMSrmm can be accessed when needed. One of the specific audits available and performed is the extracts of all references to tape from DFSMShsm CDSs (except ABARS tape) and compared to the DFSMSrmm CDS.

Other DFSMSrmm diagnostics are included:

- Finding tapes in the tape table of contents (TTOC) and MCT/MCD, but not in DFSMSrmm
- Finding tapes that are permanent in DFSMSrmm, but not in TTOC
- Finding tapes that are in DFSMSrmm, valid in DFSMShsm, but no entries are valid entries
- Finding tapes that are in DFSMShsm but expired in DFSMSrmm
- Finding tapes that are active DFSMSrmm scratch status

The audits are performed by the submission of a single batch job. This batch job performs multiple standard equivalent DFSMShsm command audits without negatively affecting system resources, DFSMSrmm, or your DFSMShsm environments.

The key to finding the errors is to verify and report these errors accurately. Mainstar's Tape Audit accomplishes this by performing a direct catalog lookup for every instance in which an error is located. The product is bidirectional about finding, verifying, and reporting errors. When the errors are verified, Mainstar *Fast*Audit/390 provides an ISPF interface for easy and organized viewing of the errors and files in question. The *Fast*Audit/390 product is shipped with easy-to-use and understand fixes to assist in correcting the errors.

The client has the ability to create, save, and submit custom fixes, too. All errors, discrepancies, fixes, and output are easily accessible and viewed through the product ISPF interface panel for future reference.

For more information about *Fast*Audit/390, visit Mainstar's website:

http://www.rocketsoftware.com

7.10.5 Using DFSMSrmm with BTLS

You can also use DFSMSrmm to manage volumes in any automated tape library that has special software, including an IBM Tape Library Dataserver that is managed using Basic Tape Library Support (BTLS).

Both system-managed tape and BTLS support the use of IBM 3494 and 3495 Automated Tape Library Dataservers. DFSMSrmm is integrated with system-managed tape but has no direct interface with BTLS.

You can use DFSMSrmm to work with BTLS to perform these tasks:

- Select volumes from scratch pools
- Return volumes to scratch status

Selecting a scratch pool

BTLS provides a way of selecting from any one of its eight scratch pools using job names. If you already use the data set name to select a scratch pool, you can use DFSMSrmm to select the scratch pool, and DFSMSrmm tells BTLS which pool is to be used. Use the DFSMSrmm VLPOOL command in PARMLIB to associate an DFSMSrmm scratch pool with the name of a BTLS scratch pool. For example, you can have the following scratch pool definitions, as shown in Example 7-24.

Example 7-24 Scratch pool definitions

```
VLPOOL PREFIX(10*) TYPE(S) MEDIANAME(CARTS) NAME(SCRTCH1) -
       DESCRIPTION('BTLS pool 1') EXPDTCHECK(N)
VLPOOL PREFIX(2*) TYPE(S) MEDIANAME(3590) NAME(SCRTCH7) -
       DESCRIPTION('BTLS pool 7') EXPDTCHECK(N)
VLPOOL PREFIX(*) TYPE(S) MEDIANAME(TAPE) -
       DESCRIPTION('default pool') EXPDTCHECK(N)
```

To use the data set name to select the scratch pool, you must implement the EDGUX100 installation exit. The *DFSMSrmm Installation and Customization Guide,* SC26-7405, provides details about how to customize, install, and use this exit.

When BTLS issues a mount to the library for a nonspecific volume, DFSMSrmm uses the IGXMSGEX installation exit to return the correct BTLS scratch pool name to BTLS, and BTLS uses that name instead of any pool it might have selected.

Returning volumes to scratch status

Although BTLS tracks the status of volumes, changing them to private status when they are used for a nonspecific mount, it provides no method of deciding when to return the volume to scratch. You use your tape management system to do this. DFSMSrmm provides simple TSO commands that you can use to drive the BTLS LIBRARY command to change a volume back to scratch status. When you have run DFSMSrmm inventory management, you can generate lists of volumes in scratch status and use them as input to the BTLS LIBRARY command to update BTLS volume status. See Example 7-25 for details.

Example 7-25 BTLS status synchronization

```
//SYNCBTLS PROC
//TMP      EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//*
//AMS      EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//LIBIN    DD   DISP=SHR,DSN=userid.EXEC.RMM.CLIST
//         PEND
//S1      EXEC SYNCHBTLS
//TMP.SYSTSIN DD *
RMM SEARCHVOLUME VOLUME(10*) OWNER(*) LIMIT(*) STATUS(SCRATCH) -
NOLIST CLIST
/*
//AMS.SYSIN DD *
LIBRARY SETCAT UNIT(123) CATEGORY(SCRTCH1)
/*
//S7       EXEC SYNCHBTLS
//TMP.SYSTSIN DD *
RMM SEARCHVOLUME VOLUME(2*) OWNER(*) LIMIT(*) STATUS(SCRATCH) -
NOLIST CLIST
/*
//AMS.SYSIN DD *
```

```
LIBRARY SETCAT UNIT(123) CATEGORY(SCRTCH7)
/*
//SM EXEC SYNCHBTLS
//TMP.SYSTSIN DD *
RMM SEARCHVOLUME VOLUME(*) OWNER(*) LIMIT(*) STATUS(MASTER) -
LOCATION(SHELF) INTRANSIT(N) NOLIST CLIST
/*
//AMS.SYSIN DD *
LIBRARY SETCAT UNIT(123) CATEGORY(PRIVATE)
/*
//EJECT EXEC SYNCHBTLS
//TMP.SYSTSIN DD *
RMM SEARCHVOLUME VOLUME(*) OWNER(*) LIMIT(*) DEST(VAULT1) -
LOCATION(SHELF) INTRANSIT(Y) NOLIST CLIST
/*
//AMS.SYSIN DD *
LIBRARY SETCAT UNIT(123) CATEGORY(XEJECTB)
/*
```

By customizing the RMM TSO command, you can select volumes from a single scratch pool if necessary so that BTLS only returns a small number of volumes to scratch status. Similarly, you can use DFSMSrmm commands to build eject lists.

7.10.6 Using DFSMSrmm with non-IBM libraries

Whatever tape library is installed in your system, DFSMSrmm can support it through a user interface that tells the library which volumes to return to scratch status and which volumes must be ejected and moved to another storage location.

With the DFSMSrmm reporting and command capabilities, it is simple to build this interface with REXX, DFSORT, DFSMSrmm utilities, and RMM TSO subcommands.

You must analyze what the vendor tape library requires as input for retention and movement processing and then customize DFSMSrmm to provide the library with the correct input.

Ensure that you keep the vendor tape library catalog and the DFSMSrmm database synchronized to avoid any mismatch that will cause the rejection of tape mounts.

STK tape libraries

This section provides a simple interface example between an STK Silo tape library and DFSMSrmm to eject tapes that need to be moved to another location. After running DFSMSrmm inventory management (storage location management processing DSTORE), you can create a list of all volumes that are moving by using DFSMSrmm commands. Follow these steps:

1. Use the DFSMSrmm commands as shown in Figure 7-59.

```
RMM SV VOLUME(*) DESTINATION( LOCAL) CLIST('      ','  -')
RMM SV VOLUME(*) DESTINATION( REMOTE) CLIST('      ','  -')
RMM SV VOLUME(*) DESTINATION( DISTANT) CLIST('      ','  -')
RMM SV VOLUME(*) DESTINATION( my_loc) CLIST('      ','  -')
```

Figure 7-59 Using RMM SEARCHVOLUME subcommands

Note: Unless you exploit the capability of APAR OW30790 and use a pre-allocated RMMCLIST DD name, you can use only one RMM TSO subcommand at a time. This is because each call to the CLIST option overwrites the data that was created previously.

The resulting list, which is stored in the RMMCLIST data set, or the data set prefix.EXEC.RMM.CLIST, consists of VOLSERs only. The data set uses blocked, variable-length records, with a logical record length of 255 if you have not specified any other data control block (DCB) information. The prefix is the TSO user PROFILE PREFIX that the user has currently defined. If you have specified NOPREFIX, DFSMSrmm allocates a data set using the RACF user ID or the jobname to avoid using EXEC.RMM.CLIST.

If you are using the RMM TSO subcommands in a batch job, you can set the prefix in front of the RMM TSO subcommand.

2. Convert the list from variable-length records to fixed-length records. HSC/MVS control statement syntax has the following convention: the control statement for each utility program consists of a command (indicating the utility function) followed by parameters, as applicable, in 80-character card-image records (see the *HSC/MVS Programmer's Guide*, *Volume 2* that is provided with the product).
3. Edit this list, add an EJECT statement in front, and close the command. Optionally, you can specify a CAP statement as shown in Figure 7-60.

```
EJECT VOLSER( -
      volser1 -
      volser2 -
      volser3 -
             -)
CAP( cap-list )
```

Figure 7-60 Adding the EJECT command to the volume list

Each single EJECT command causes an operator interaction.

4. Run the HSC update utility, SLUADMIN (see the *HSC/MVS System Programmer's Guide, Volume 1* that is provided with the product). The HSC Initialize Cartridge Utility (SLUADMIN) in Example 7-26 shows a sample JCL that you can use to initialize cartridges with the HSC SLUADMIN utility by using DFSMSrmm EDGINERS processing.

Example 7-26 JCL for initializing cartridges with EDGINERS

```
//TINIT    EXEC PGM=SLUADMIN,TIME=1440,PARM=MIXED
//TAPE     DD   UNIT=(580,1,DEFER),DISP=NEW
//* EDGINERS TAPE UNIT DD
//SLSTAPE  DD  UNIT=AFF=TAPE INIT CARTRIDGES INIT TAPE
//* UNIT DD.
//SYSIN    DD   UNIT=VIO,SPACE=(TRK,1) EDGINERS CONTROL DATASET
//SLSPRINT DD   SYSOUT=* UTILITY MESSAGES
//SYSPRINT DD   SYSOUT=* EDGINERS MESSAGES
//SLSINIT  DD   *
   INIT LABEL(SL) VOL(******)
//*
//SLSIN    DD   *
* WHEN USING THE DFSMSRMM TAPE INITIALIZATION PROGRAM EDGINERS, YOU
* MUST SPECIFY THE 'PROGRAM' PARAMETER ON THE INIT CARTRIDGES
* CONTROL STATEMENT.
 INITIALIZE CAP(000) PROGRAM(EDGINERS) OPTION(SCRATCH) CNTLDD(SYSIN)
```

```
/*
```

The HSC scratch conversion SLUCONDB program is provided to generate scratch transactions from the DFSMSrmm tape management system. SLUCONDB calls the routine SLUDRRMM to read the DFSMSrmm report extract file if RMM is specified in the EXEC JCL statement. This program uses the DFSMSrmm EDGRVEXT macro to map the layout of the extract file. The following list shows the fields that are used:

- RVTYPE
- RVVOLSER
- RVLCDATE
- RVNAME
- RVSTATUS
- RVEXPDT
- RVEXPDTO
- RVLABEL

SLUDRRMM processes the extract and passes the volume record information to SLUCONDB, which builds scratch card images to input to SLUADMIN. Volume status scratch (scratch or non-scratch) is then updated in the HSC CDS for each volume record in the extract file. Dates on the DFSMSrmm report must be in the Julian Date format (EDGHSKP run with PARM 'DATEFORM(J)'). Tapes listed in the DFSMSrmm report without expiration dates are skipped by SLUDRRMM. Example 7-27 shows the DFSMSrmm and HSC synchronization JCL.

Example 7-27 DFSMSrmm and HSC synchronization JCL

```
/S1       EXEC PGM=EDGHSKP,PARM='RPTEXT,DATEFORM(J)'
//MESSAGE  DD   DSN=RMM.MESSAGE.DATASET,DISP=SHR
//REPTEXT  DD   DSN=RMM.REPORT.EXTRACT.HSC.ONLY,DISP=SHR
//SYSIN    DD   *
 RPTEXT RECORDS(V)
/*
//S2       EXEC PGM=SLUCONDB,
//         PARM=('RMM,SCRPOOL(SL)',,MIXED')
//SLSTMS   DD   DSN=RMM.REPORT.EXTRACTHSC.ONLY,DISP=SHR
//SLSSOUT  DD   DSN=&&SCUPINPT,DISP=(NEW,PASS),UNIT=,SPACE=DCB=
//SLSPRINT DD   SYSOUT=*
//S3       EXEC PGM=SLUADMIN
//SLSIN    DD   DSN=&&SCUPINPT,DISP=(OLD,DELETE)
//SLSPRINT DD   SYSOUT=*
```

The SLUCONDB utility, when PARM=RMM is specified, checks in the DFSMSrmm extract file to see if the MEDIANAME of the tapes being returned to scratch is either 3480, 3490, or 3590. For example, if you changed the default media name 3480, and you are using CARTS, you never get your tapes selected as scratch by the utility.

Note: To save space and to speed up the SLUCONDB process, you can create an extract file with only V records.

Update the MEDIANAME used in the STK SLUADMIN program by updating the SLUDRRMM source module. After that, compile and link edit it.

Example 7-28 on page 278 shows the original program code.

Example 7-28 Original SLUDRRMM macro

```
......
READRCD  EQU   *
         LA    R2,BUFFER2+4              LOAD ADRESS OF INPUT BUFFER
         GET   INPUT,BUFFER2             READ RECORD
         USING RVEXT,R2                  ESTABLISH ADRESSABILITY TO RCD.
         CLI   RVTYPE,C'V'               WAS IT A VOLUME RECORD?
         BNE   READRCD                   NO - READ NEXT RECORD.
         CLC   RVNAME,=CL8'3480'         WAS RECORD FOR A 3480 VOL?
         BE    READCONT                  YES - CHECK STATUS.
         CLC   RVNAME,=CL8'3490'         WAS RECORD FOR A 3490 VOL?
         BNE   READRCD                   NO - READ NEXT RECORD.
         CLC   RVNAME,=CL8'3590'         WAS RECORD FOR A 3490 VOL?
         BNE   READRCD                   NO - READ NEXT RECORD.
READCONT CLC   RVSTATUS,=CL8'STATUS'     WAS RECORD IN SCRATCH STATUS?
         BNE   READRCD                   NO - READ NEXT RECORD.
```

The modified program code is shown in Example 7-29.

Example 7-29 Modified SLUDRRMM macro

```
......
READRCD  EQU   *
         LA    R2,BUFFER2+4              LOAD ADRESS OF INPUT BUFFER
         GET   INPUT,BUFFER2             READ RECORD
         USING RVEXT,R2                  ESTABLISH ADRESSABILITY TO RCD.
         CLI   RVTYPE,C'V'               WAS IT A VOLUME RECORD?
         BNE   READRCD                   NO - READ NEXT RECORD.
         CLC   RVNAME,=CL8'3480'         WAS RECORD FOR A 3480 VOL?
         BE    READCONT                  YES - CHECK STATUS.
         CLC   RVNAME,=CL8'CARTS'        WAS RECORD FOR CARTRIDGES?
         BE    READCONT                  YES - CHECK STATUS.
         CLC   RVNAME,=CL8'3490'         WAS RECORD FOR A 3490 VOL?
         BE    READCONT                  YES - CHECK STATUS.
         CLC   RVNAME,=CL8'3590'         WAS RECORD FOR A 3590 VOL?
         BNE   READRCD                   NO - READ NEXT RECORD.
READCONT CLC   RVSTATUS,=CL8'STATUS'     WAS RECORD IN SCRATCH STATUS?
         BNE   READRCD                   NO - READ NEXT RECORD.
```

Note: Define in the routine all the MEDIANAMES that you have specified in the VLPOOL command. In this example, you have used the MEDIANAME 3480, 3490, 3590, or CARTS in different VLPOOL commands.

STK provides a sample insert/delete user exit SLSUX06. This exit enables the HSC to get control when the database is being updated to add or delete a volume. For details about how to tailor this exit to work with IBM DFSMSrmm, see the STK tape library documentation.

VTCS to DFSMSrmm interface

There is a communication between Virtual Tape Control System (VTCS) and DFSMSrmm to bypass the DFSMSrmm HDR1 checking. Before VSM Patch-ID# 131603-01 was introduced, the VTCS issued an RMM CHANGEVOLUME subcommand to clear out the HDR1 information in the DFSMSrmm control data set. Now, the VTSS sets the HDR1 data set name to 'STK VSMSCRTCH' when it mounts a scratch tape. For more information, see “Using SCRTCH tapes without HDR1 DSN information”.

You can enable, disable, or check the VTCS to DFSMSrmm interface. The interface is automatically disabled at VTCS initialization but you can change the status at any time. Figure 7-61 shows the command to manually enable the VTCS to DFSMSrmm interface.

```
▶▶──SET─RMM──────────────────────────────────────────────────────────────────◀◀
                 │─ENABLE────────│
                 │─DISABLE───────│
                 │───────────────│
```

Figure 7-61 VTCS SET RMM command syntax

The following explanations apply to Figure 7-61:

ENABLE This optional parameter enables the VTCS to DFSMSrmm interface. With the interface enabled, VTCS will communicate with DFSMSrmm on all scratch mounts and scratch requests of Virtual Tape Volumes (VTVs) whose MGMTCLAS specifies DELSCR (YES).

DISABLE This optional parameter disables the VTCS to DFSMSrmm interface. The interface is disabled automatically at VTCS initialization.

blank Entering this command without the ENABLE or DISABLE parameter will display the current setting of the interface.

Using SCRTCH tapes without HDR1 DSN information

Message EDG4026I is issued for scratch tapes mounted for any virtual tape product because no data set name is provided in the tape HDR1 label. This problem only occurs when the virtual tape product does not issue the subcommand shown in Figure 7-62 via the application programming interface (API) at scratch mount time. In certain configurations, it is possible that the subcommand cannot be issued. Your virtual tape product vendor must make a change to support the special recognition made by RMM and avoid the CHANGEVOLUME subcommand.

```
RMM CHANGEVOLUME volser INIT(Y) CRLSE(INIT)
```

Figure 7-62 RMM CHANGEVOLUME subcommand

With APAR OA09171, DFSMSrmm checks for the special character strings in the System Code field of the HDR1 to enable the virtual tape product to confirm that no HDR1 DSNAME checking is to be performed by DFSMSrmm. Figure 7-63 shows the two special character strings that can be used to bypass HDR1 checking.

```
STKVSM   DC    CL13'STK VSMSCRTCH'
ANYVSM   DC    CL13'VirtualSCRTCH'
```

Figure 7-63 Special character string to bypass HDR1 checking

8

System-managed library support

In this chapter, we describe the different methods Data Facility System Managed Storage removable media manager (DFSMSrmm) supports for partitioning your system-managed tape libraries.

In this chapter, we discuss the following enhancements:

- Storage management subsystem-managed tape library overview
- Sharing a library between multiple systems
- System-managed library partitioning
- Partitioning support
- Using PRTITION and OPENRULE commands
- Converting REJECT commands
- Conversion from REJECT to PRTITION and OPENRULE

By the end of this chapter, you will understand how you can partition your IBM system-managed tape libraries using the CBRUXENT and EDGUX200 user exits, the REJECT command or the PRTITION and OPENRULE commands, and how you can convert your REJECT commands.

8.1 Storage management subsystem-managed tape library overview

A system-managed tape library consists of tape volumes and tape devices that are defined in the tape configuration database (TCDB). The tape configuration database is an integrated catalog facility user catalog marked as a volume catalog (VOLCAT) containing tape volumes and tape library records. A system-managed tape library can be either automated or manual.

You can have several automated tape libraries (ATLs) or manual tape libraries. You use an installation-defined library name to define each automated tape library or manual tape library to the system. DFSMSrmm treats each system-managed tape library as a separate location or destination.

A storage management subsystem (SMS)-managed tape library can be shared by multiple operating system platforms (open systems and IBM System z® attached hosts) and can be partitioned across multiple System z hosts.

8.2 Sharing a library between multiple systems

A storage management subsystem (SMS)-managed tape library can be shared among multiple systems and, in some cases, among multiple SMS complexes. In all cases, it is the responsibility of the installation staff to ensure that an individual library-resident tape drive is not allocated by two systems concurrently. This means that the tape drive can be online to only one SMS complex.

There must be a single shared tape configuration database (TCDB) among all systems in all SMS complexes. This means there is one general volume catalog, and at the most, one specific volume catalog for each valid initial VOLSER character.

Figure 8-1 shows a full shared automated tape library environment with a single DFSMSrmm control data set (CDS), a single tape configuration database (TCDB), and a single RACF database. All ICF user catalogs are fully shared.

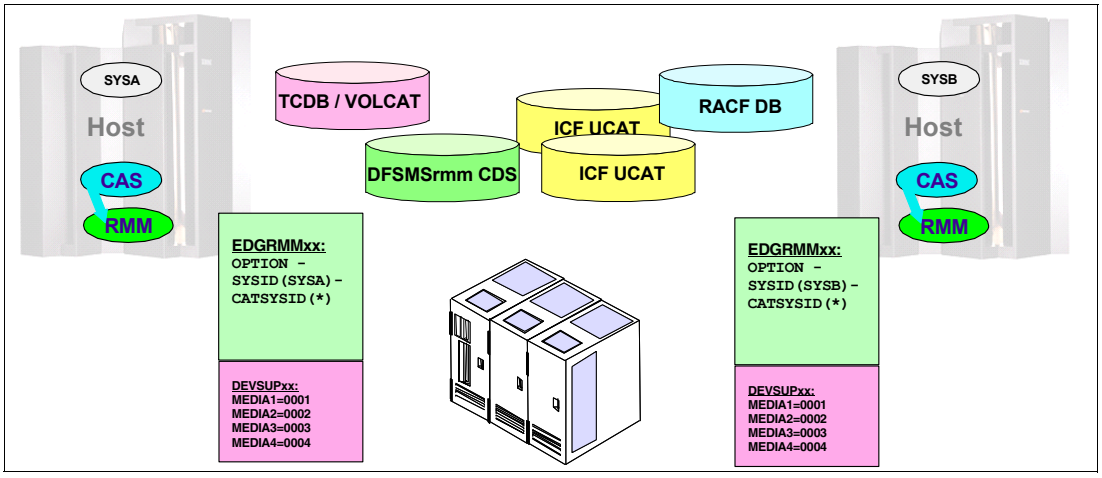


Figure 8-1 A full shared tape library environment

In this shared tape environment, you can perform these tasks:

- Run the expiration processing on any system
- Insert new cartridges without the need to pre-define them in the TCDB or in the DFSMSrmm CDS
- Implement catalog synchronization easily
- Use all volumes on any system

8.3 System-managed library partitioning

Depending on how you want to partition your library, there are different methods:

- Modifying the CBRUXENT and EDGUX200 user exits
- Using the USE operand of the RMM ADDVOLUME subcommand if a volume is added to the DFSMSrmm CDS
- Specifying the REJECT command in the EDGRMMnn parmlib member
- Specifying the OPENRULE and PRTITION commands in the EDGRMMnn parmlib member

In any case, you need to use the DEVSUPxx parmlib member to set different scratch volume categories. You must specify a volume category for each different media type and also for volumes in PRIVATE status.

8.3.1 Cartridge insert process

When a cartridge enters the tape library, it might be accepted and assigned a corresponding category by a system, or it might be rejected depending on the partition configuration. Figure 8-2 on page 284 shows the procedure a system uses to decide to accept or reject a volume.

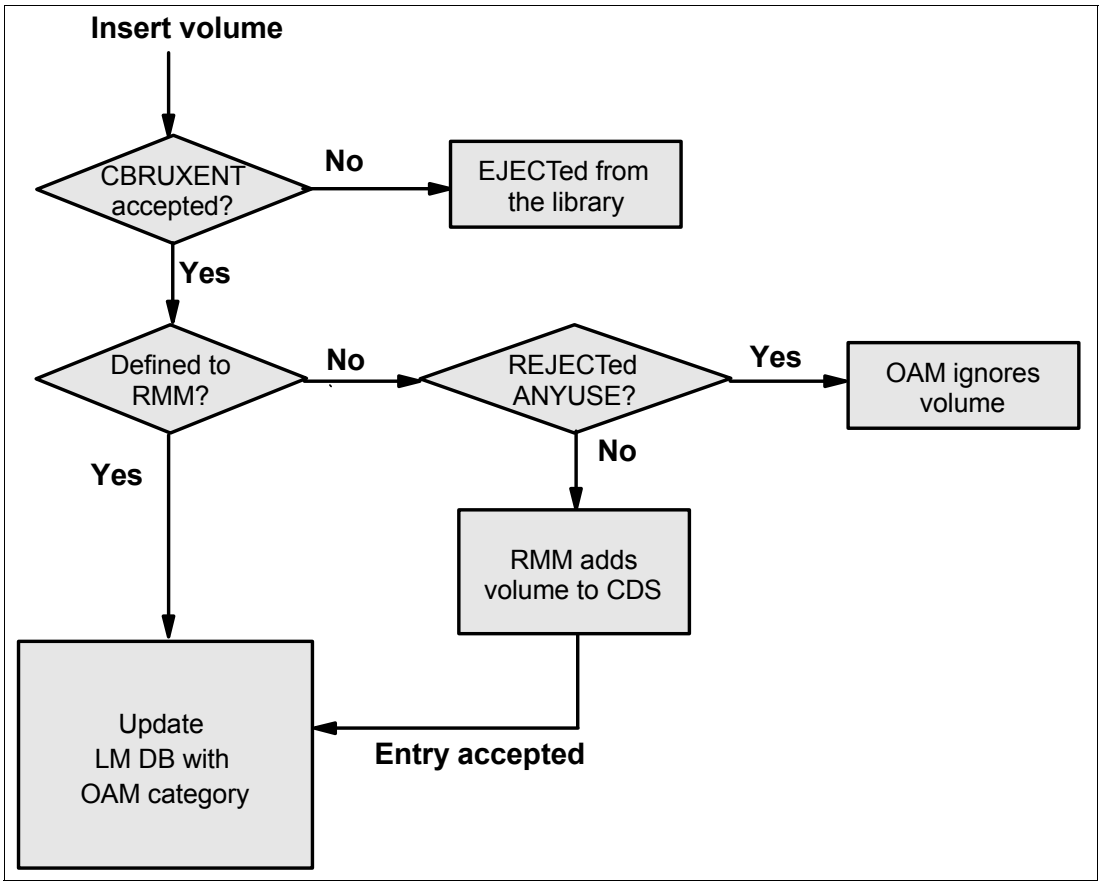


Figure 8-2 Volumes enter a tape library

Note: When you change the CBRUXENT exit, you need to assemble and link edit the module after each maintenance.

With z/OS 1.10 and later, new parmlib controls are provided to enable simplified and more powerful partitioning of system-managed libraries and control of application use of tape volumes. Now, you can use PRTITION and OPENRULE commands in the EDGRMM parmlib member to control library partitioning and the use of volumes by applications.

Important: If you use REJECT commands, you have to convert from the use of REJECT commands in order to use the PRTITION and OPENRULE commands.

You can partition a system-managed library, including a Virtual Tape Server (VTS), by performing these tasks:

- Specify the USE operand value on the RMM ADDVOLUME or RMM CHANGEVOLUME subcommands. You can set this value to MVS, VM, or both. If you do not specify MVS for a volume, DFSMSrmm prevents the volume from being defined in the volume catalog on this system.

- Through the use of the PRTITION and OPENRULE parmlib commands, you can simplify the maintenance of the parmlib members as your libraries and volume ranges change. Operands on the OPENRULE and PRTITION commands allow global actions to be set. You can use one or more specific overrides based on volume sets that have different requirements. Typically, you can add a new range of volumes for use by a single partition and only that one system will need to be updated. The OPENRULE and PRTITION commands allow you to define whether they apply to volumes defined to DFSMSrmm or not. You can use operands on the OPENRULE command to automatically ignore volumes, rather than using EXPDT=98000, ACCODE=xCANORES, or a customized EDGUX100.
- Define parmlib member EDGRMMxx REJECT prefixes. You can use REJECT to prevent a volume not defined to DFSMSrmm from being defined in a system-managed tape library. The REJECT ANYUSE(prefix) operand prevents a volume from being defined in the system-managed tape library on the current system. The REJECT OUTPUT(prefix) operand allows you to define the volume to the system-managed tape library but only use the volume for input processing.

8.3.2 The sequence of processing

The processing path that is taken depends on whether PRTITION or OPENRULE or REJECT parmlib commands are used. If REJECT is used, but PRTITION or OPENRULE is not used, processing is unchanged. If you have specified PRTITION and OPENRULE parmlib operands, the processing follows this sequence:

- IGNORE processing determines whether this request will be ignored or processed by DFSMSrmm
- Partitioning checking using PRTITION
- REJECT processing based on OPENRULE

8.3.3 Partitioning a library between MVS and other systems

You can use DFSMSrmm to manage all your volumes. MVS volumes are managed automatically, and volumes from other systems are managed manually. Each time you add a volume, specify whether this volume will be used from MVS or from another system by using the TSO ADDVOLUME subcommand operand that is shown in Figure 8-3.

```
RMM AV volser USE(MVS)
or
RMM AV volser USE(VM)
or
RMM AV volser USE(IRMM)
```

Figure 8-3 RMM ADDVOLUME subcommand with the USE attribute

Figure 8-4 shows how you can add the VM0001 volume to be used only from a VM system.

```
RMM ADDVOLUME VM0001 DENSITY(3480) OWNER(YCJRES)-
    STATUS(USER) USE(VM) MEDIANAME(3480)
```

Figure 8-4 RMM ADDVOLUME subcommand for a volume used on a VM system

MVS uses only its own volumes, and DFSMSrmm manages these volumes automatically.

If you prefer, DFSMSrmm has information about the status of the VM volumes; you need to change this information manually with commands.

8.3.4 Partitioning a library by CBRUXENT and EDGUX200 user exits

The scenario is shown in Figure 8-5. One library and two systems are sharing the same CDS and the same TCDB.

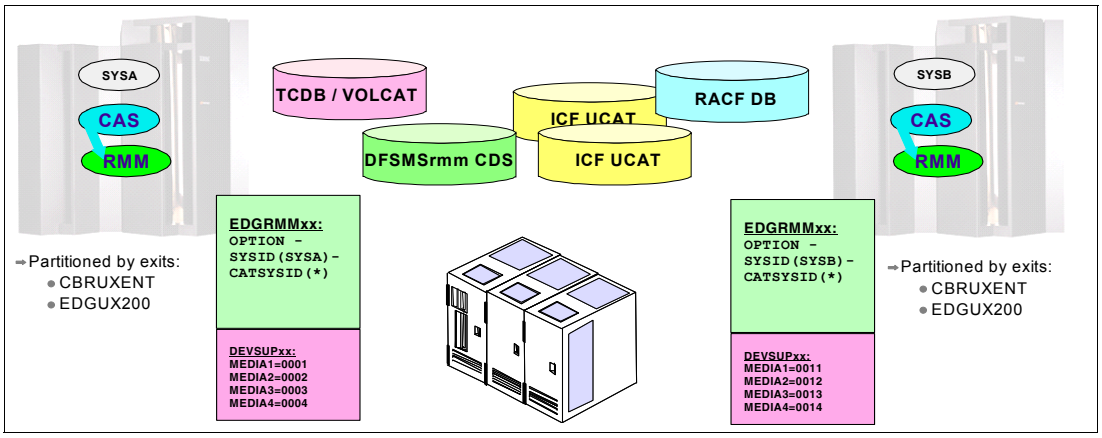


Figure 8-5 Scenario with one CDS and one TCDB

Both the TCDB and CDS are shared between systems; you need to modify the CBRUXENT and EDGUX200 exits to partition the library. When a volume that was moved out of the library comes back again because a record for this volume exists in the CDS, the REJECT command in EDGRMMxx does not work.

This kind of partition offers you a convenient way to share private volumes among systems for all the volume entries that are in the same TCDB. No additional consideration is introduced by this partition, because both CDS and TCDB contain the information for all SMS volumes.

You need to run EXPROC processing on each system name, and use a customized EDGUX200 exit to prevent volumes from returning to scratch on the wrong system. The decisions about which volumes to process must match those coded in the customized CBRUXENT exit.

8.3.5 Partitioned library with shared CDS and TCDB

In the first example in Figure 8-6 on page 287, we show you a system-managed tape library partitioned by these methods:

- Using CBRUXENT user exits on each system
- Using the EDGUX200 user exit on each system
- Having different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system, but all volumes in the PRIVATE category are shared between the two systems.

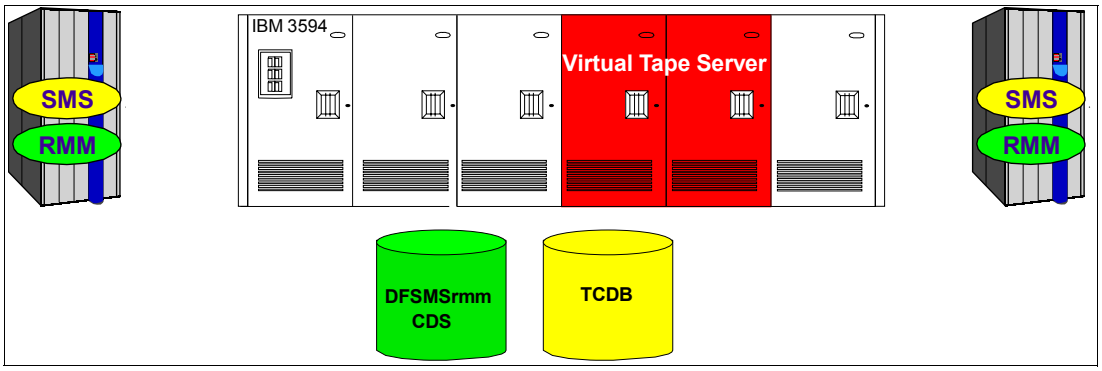


Figure 8-6 Partitioned library with shared CDS and TCDB

8.3.6 Partitioned library with shared TCDB and different CDSs

In the example in Figure 8-7, we show you a system-managed tape library partitioned by these methods:

- A different DFSMSrmm control data set for each system
- Use of REJECT ANYUSE in the EDGRMMnn parmlib member
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by CDS, but all volumes in the PRIVATE category are shared between the two systems. To access these volumes, you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 user exit is needed.

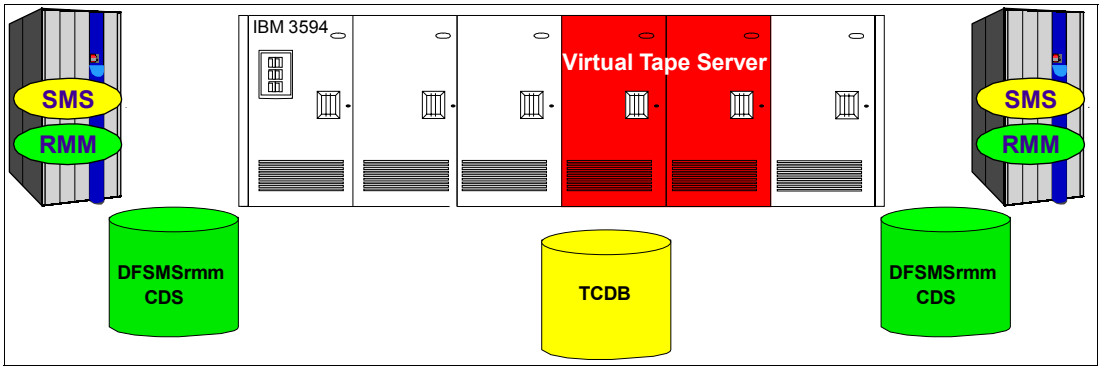


Figure 8-7 Partitioned library with shared TCDB and different CDSs

8.3.7 Partitioned library with unshared CDS and TCDB

In the third example shown in Figure 8-8 on page 288, you can see a system-managed tape library partitioned by these methods:

- System
- Use of REJECT ANYUSE in the EDGRMMnn parmlib member
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system. All volumes are not shared so you have to define a volume in private status in the TCDB first. To access this volume, you have to specify one of the two JCL parameters: EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 user exit is needed.

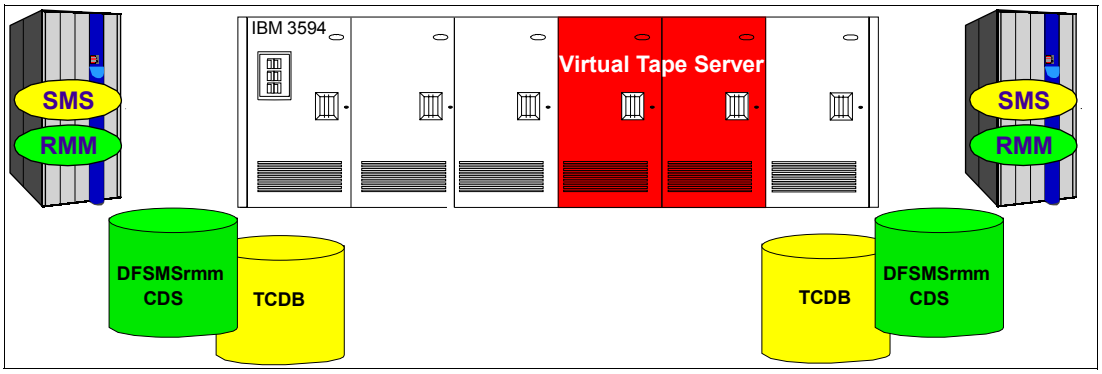


Figure 8-8 Partitioned library with unshared CDS and TCDB

All volumes are not shared so you have to define a volume in private status in the TCDB before you can use it. Figure 8-9 shows how you can add a volume in the TCDB using access method services (AMS) commands. To access this volume, you have to specify one of the two JCL parameters: EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 user exit is needed.

```
//STEP0000 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
   CREATE VOLUMEENTRY (NAME(VLZTB82) -
   LIBRARYNAME(ATV3494J) -
   LOCATION(LIBRARY) -
   USEATTRIBUTE(PRIVATE) -
   STORAGEGROUP(ATV3494J) -
   MEDIATYPE(MEDIA2)
/*
```

Figure 8-9 Add and delete a volume entry to the TCDB

After you have successfully processed the volume you have previously defined in the TCDB, delete the entry from the TCDB by using the command shown in Figure 8-10.

```
//STEP0000 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
 DELETE ('VLZTB82') VOLUMEENTRY PURGE
/*
```

Figure 8-10 Delete a volume entry from the TCDB

8.3.8 Partitioned library with shared CDS and unshared TCDB

Figure 8-11 on page 289 shows a system-managed library partitioned in these ways:

- By system
- By using a customized version of CBRUXENT user exits on each system
- By different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system based on the TCDB. All volumes are not shared so you have to define a volume in private status in the TCDB first before you can use it. Figure 8-9 on page 288 shows how you can add a volume in the TCDB using AMS commands. After you have successfully processed the volume you have previously defined in the TCDB, delete the entry from the TCDB by using the command as shown in Figure 8-10 on page 288.

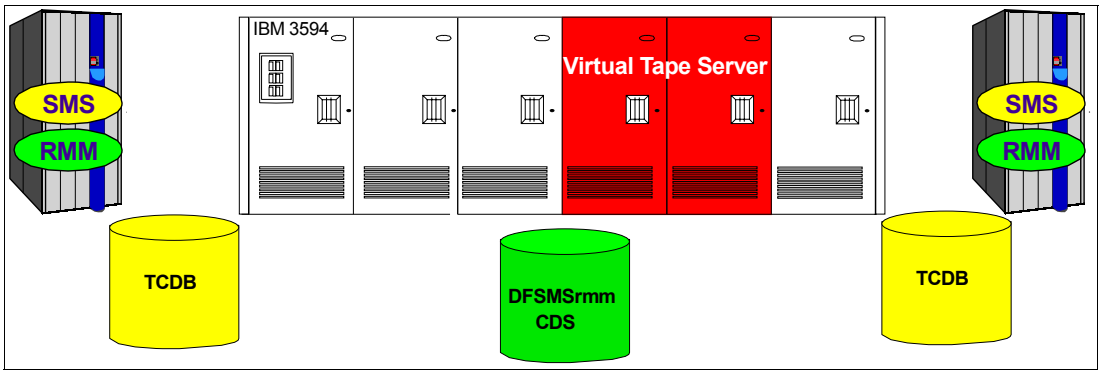


Figure 8-11 Partitioned library with shared CDS and unshared TCDB

8.4 Partitioning support

Each system in the RMMplex uses its own parmlib options to determine the processing that is required. For releases before z/OS V1R10, the REJECT commands are used for both partitioning and for Open/Close/EOV volume use decisions. On z/OS V1R10 and later releases, the REJECT commands are only used if specified and neither PRTITION nor OPENRULE commands are defined. When you define REJECT commands and PRTITION or OPENRULE commands, the REJECT commands cause an DFSMSrmm start-up error and you must reply to EDG0239E to continue the start-up or restart DFSMSrmm with a corrected parmlib. If there are no REJECT commands, the PRTITION and OPENRULE command defaults are used. EDG0239E is described:

EDG0239E `REJECT COMMAND FOUND IN PARMLIB AND NO LONGER SUPPORTED`

Explanation This message is issued during initialization when a REJECT command has been found, but one or more PRTITION and OPENRULE commands are also specified. You must use either REJECT commands or OPENRULE and PRTITION commands.

System action DFSMSrmm initialization stops. This message is followed by message EDG0215D.

Operator response Notify the system programmer. Reply to message EDG0215D as directed. Restart DFSMSrmm when the system programmer has corrected the error.

System programmer response
Do not attempt to use both REJECT commands and PRTITION or OPENRULE commands in parmlib. If you are implementing PRTITION and OPENRULE commands, you must remove the REJECT commands from parmlib. **You can advise the operator to reply Y to message EDG0215D to continue and ignore the REJECT commands**.

Source DFSMSrmm

Detecting module EDGPARM

Routing code 3

Descriptor code 3

8.5 Using PRTITION and OPENRULE commands

Using the PRTITION commands, you can control partitioning entry/insert, export/import, eject, and Common User Access (CUA) processing for system-managed volumes:

- DFSMSrmm partitioning is based on a global setting that you can change. The default is that all system-managed and non-system-managed volumes are accepted. This can be represented by the command shown in Figure 8-12.

```
PRTITION VOLUME(*) TYPE(ALL) SMT(ACCEPT) NOSMT(ACCEPT LOCATION(SHELF))
```

Figure 8-12 Partitioning based on global setting

You can change the global setting so that all volumes are ignored by using the command shown in Figure 8-13.

```
PRTITION VOLUME(*) TYPE(ALL) SMT(IGNORE) NOSMT(IGNORE)
```

Figure 8-13 Partitioning based on global setting ignoring all volumes

- The global command shown in Figure 8-14 applies to volumes not defined to DFSMSrmm.

```
PRTITION VOLUME(*) TYPE(NORMM) SMT(IGNORE) NOSMT(IGNORE)
```

Figure 8-14 Global rejecting all undefined volumes

This global command will perform the following actions:

  - All system-managed volumes that are undefined in the DFSMSrmm CDS are left in the insert category to be accepted by another system and non-system-managed volumes are not added automatically to the CDS.

    Note: This command replaces the REJECT ANYUSE(*) definition.

  - The remainder of the default command [PRTITION VOLUME(*) TYPE(RMM) SMT(ACCEPT) NOSMT(ACCEPT)] that is not overridden by your global commands in parmlib is used to handle TYPE(RMM) volumes.
  - With this approach, you must predefine system-managed volumes to DFSMSrmm to enable ownership of volumes during entry/insert processing.

- No PRTITION commands. When REJECT commands are used, processing is as for earlier releases unless any OPENRULE statements are defined. In the latter case and when no REJECT commands are defined, the defaults are used for partitioning.
- Using selective PRTITION statements on top of a global command, you can be specific about which volumes will be owned by the current system/partition.

- All accepted volumes (ACCEPT) that are not defined to DFSMSrmm are added automatically to the DFSMSrmm CDS and owned by this system/partition, unless it is eject processing:
  - For system-managed volumes, “owned by this system/partition” means that the volume is defined to DFSMSrmm, has an entry in the TCDB, and the scratch category used by the Library Manager (LM) is set based on your values in this system’s DEVSUPxx.
  - The Interactive Storage Management Facility (ISMF) library default entry status is used for added volumes.
- Predefined volumes are only considered when the PRTITION command specifies TYPE(RMM) or TYPE(ALL). The volume status for the TCDB entry is set by DFSMSrmm from the volume information.

Note: As a result of this flexibility, you no longer need to customize the CBRUXENT exit.

8.5.1 Defining PRTITION commands

Example 8-1 shows the correct use of the EDGRMMnn parmlib member PRTITION command.

Example 8-1 EDGRMMnn PRTITION command

```
▶▶──PRTITION─────VOLUME──────(──Volser_or_prefix──)──────────────────────────▶
                 │                                │
                 │─VOLUMERANGE─(──Volser_or_prefix──)─│


     │──TYPE(ALL)──│      │──SMT(ACCEPT)──NOSMT(ACCEPT LOCATION(SHELF)──│
▶────────────────────────────────────────────────────────────────────────────◀◀
     │             │      │                                              │
     │──Selection──│      │──Action──────────────────────────────────────│


Selection:
│──────────────────────────────────────────────────────────────────────────│
   │                     │
   │        │─ALL──│     │
   │──TYPE(──────────)──│
            │─RMM──│
            │─NORMM─│

Action:
│────────────────────────────────────────────────────────────────────────────│
   │                     │  │                                              │
   │        │─ACCEPT─│   │  │        │─ACCEPT─│  │─LOCATION─(──SHELF────)│  │
   │─SMT─(──────────)─│  │─NOSMT─(──────────────────────────────────────)─│
            │─IGNORE─│              │─IGNORE─│  │                      │
                                                │           │─SHELF─│  │
                                                │─LOCATION─(──────────)│
                                                            │─Locdef─│
```

If there are no PRTITION or OPENRULE commands, or there are no REJECT commands, found in EDGRMMxx parmlib, default entries are created from the command shown in Figure 8-15 to cover any volumes for which you do not define a PRTITION command.

```
PRTITION VOLUME(*) TYPE(ALL) SMT(ACCEPT) NOSMT(ACCEPT LOCATION(SHELF))
```

Figure 8-15 Default values of the PRTITION command

The following explanations apply to Figure 8-15:

SMT/NOSMT(action) Use the SMT/NOSMT operands to select the action you want based on whether the volume is system-managed or not. You can specify a different action for NOSMT and SMT volumes. System-managed volumes are identified in the following ways:

- Any request made through the EDGLCSUX programming interface that is used from the object access method (OAM) installation exits: CBRUXCUA, CBRUXENT, CBRUXEJC, and CBRUXVNL.
- During OPEN processing, the tape drive is identified as being in a system-managed library.
- During EXPROC processing, the volume is in the TCDB, or the current location, or the home location is a system-managed library.
- In any other case, the volume is treated as non-system managed.

ACCEPT/IGNORE Use this operand to identify the action to be taken by DFSMSrmm. The action applies to library entry/insert/import, eject/export, and CUA processing, OPEN processing, and EXPROC processing. During OPEN processing, once the decision is made about using IGNORE for the volume, the PRTITION checking occurs first, and the action taken before the OPENRULE REJECT/ACCEPT processing occurs. As a result, the action taken by PRTITION can influence the TYPE processing by OPENRULE.

ACCEPT DFSMSrmm processes the volume. During entry/insert import/export, CUA, and OPEN processing, the volume is added to the DFSMSrmm CDS if not yet defined. For NOSMT volumes, the location from the LOCATION operand is used to set the volume current and home locations. During EXPROC processing, the volume is processed for return to scratch. Note that a volume identified as TYPE(NORMM) with an action of ACCEPT, that is added to the CDS, now becomes a TYPE(RMM) volume for subsequent processing including OPENRULE processing. By default, all volumes are processed during EXPROC, and all volumes are accepted during library entry/insert/import or OPEN processing. During OPEN processing and during library entry/insert/import processing, any undefined volumes are automatically added to the DFSMSrmm CDS using the library default volume entry status. During OPEN processing, volumes are added to the CDS only if the current OPMODE is either WARN or PROTECT.

IGNORE

DFSMSrmm does not process the volume. During entry/insert/import processing, DFSMSrmm requests that OAM ignore the volume and leave it in the “insert” category. During CUA and eject/export processing, the request for the volume is ignored. During EXPROC processing, the return to scratch processing is skipped. During OPEN processing, the following information applies:

- For NORMM volumes, the volume is not added to the CDS.
- For RMM volumes, processing depends on the OPENRULE entries.

ACCEPT is the default value.

LOCATION(SHELF|LocdefHome)

Use this operand for NOSMT volumes that are of TYPE(NORMM) with action ACCEPT. DFSMSrmm’s add volume processing uses the defined location value as the volume’s home location and current location. You can specify either SHELF or the name of a LOCDEF defined location that has a specified TYPE(STORAGE,HOME). DFSMSrmm validates this value against the LOCDEF commands. Although the operand is only used for NORMM volumes, DFSMSrmm does not prevent you from specifying it on a PRTITION command for volumes of TYPE RMM or ALL.

The default value is SHELF.

TYPE(ALL|RMM|NORMM)

Use it to identify the type of volumes selected by the PRTITION command. The VOLSER is used to determine whether the volume is defined in the RMM CDS, and, depending on the function, the VOL1 label and HDR1 can be used. Therefore, DFSMSrmm cannot always determine whether the volume is an existing one defined to DFSMSrmm. TYPE is determined in the following manner:

- For system-managed library functions, such as import/export, insert/entry, CUA, and eject, only the VOLSER is used to determine whether the volume is defined in the RMM CDS.
- For OPEN processing, the same rules as volume ignore processing are used and the TYPE that is assigned is based on the following information:

  –TYPE(RMM):

  The volume serial number is defined in the control data set, and there is no HDR1 tape label for the volume.

  The volume serial number is defined in the control data set, and no labels are used (no label (NL), nonstandard label (NSL), or bypass label processing (BLP) with an NL tape).

  The 17 characters read from the HDR1 label of the mounted volume match the last 17 characters of the data set name in the control data set.

  –TYPE(NORMM):

  The volume serial number is not defined in the control data set.

The volume is defined, but the 17 characters read from the HDR1 label of the mounted volume do not match the last 17 characters of the data set name in the control data set.

- For EXPROC processing, TYPE(RMM) always applies because EXPROC is driven only by volumes defined to DFSMSrmm.

For each set of volumes, you can code one command with RMM and another with NORMM, but only in a single command if ALL is used:

RMM The volume must already be defined to DFSMSrmm.

NORMM The volume is not defined to DFSMSrmm.

ALL Applies to all volumes regardless of whether they are defined to DFSMSrmm. ALL is the default value.

VOLUME/VOLUMERANGE

Use these operands to select volumes that are to be managed by this command. You must specify either VOLUME or VOLUMERANGE, and each command defines a set of one or more volumes. Sets cannot overlap within a TYPE, but a set can be a subset of another and such a subset is more specific. When you code a set with TYPE(RMM) and repeat that set with TYPE(NORMM), you can specify different operands for each set. If you want to use the same operands for the sets, you can do so by coding a type of ALL.

VOLUME You can specify the volume as fully qualified or as a VOLSER prefix ending in an asterisk (*). A fully qualified volume is one to six alphanumeric, national, or special characters. A VOLSER prefix is zero to five alphanumeric, national, or special characters ending in an asterisk (*). Quotation marks are required for special characters, and the first character must not be blank. Any value ending in an asterisk (*), even if enclosed in quotation marks, is considered to be a VOLSER prefix.

VOLUMERANGE Use to select a subset of volumes based on starting and ending VOLSER. One to six characters for beginning and end of range are required to be processed, and they can be alphanumeric, national, and special characters. Quotes are required for each value regardless of the use of special characters, and the first character must not be blank. The end of the range must not be lower than the start of the range.

We take the PRTITION definitions as shown in Figure 8-16 on page 295 in our EDGRMMnn parmlib member.

```
PRTITION VOLUME(*)              /*     added 2008/03/20 NS          */-
         TYPE(NORMM)            /* undefined volumes in CDS only    */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(IGNORE)          /* ignore no-nsms managed tapes     */
PRTITION VOLUMERANGE('THM000':'THM029')  /*  added 2008/03/20 NS   */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* accept sms managed tapes         */-
         NOSMT(IGNORE)          /* ignore non-sms managed tapes     */
PRTITION VOLUME(DB*)            /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
PRTITION VOLUME(HE00*)          /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
PRTITION VOLUME(J5T*)           /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
PRTITION VOLUME(MXX*)           /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
PRTITION VOLUME(S0024*)         /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
PRTITION VOLUME(S020*)          /*     added 2008/03/20 NS          */-
         TYPE(ALL)              /* defined and undefinded volumes   */-
         SMT(IGNORE)            /* ignore sms managed tapes         */-
         NOSMT(ACCEPT)          /* accept non-sms managed tapes     */
```

Figure 8-16 EDGRMMnn PRTITION definitions

To list the current PRTITION settings, you can use the RMM TSO subcommand LISTCONTROL with the new option PRTITION as shown in Figure 8-17.

```
RMM LISTCONTROL PRTITION
or
RMM LC P
```

Figure 8-17 LISTCONTROL with option PRTITION

Figure 8-18 on page 296 shows the result of the RMM TSO subcommand LISTCONTROL. We used the option PRTITION to get only this information. Each definition is listed twice, one time to show the type RMM settings and the second time for the NORMM settings.

```
Partition Entries:
                        SMT       NOSMT
Volume or Range Type  Action Action Location
--------------- ----- ------ ------ --------
DB*             RMM   IGNORE ACCEPT SHELF
HE00*           RMM   IGNORE ACCEPT SHELF
J5T*            RMM   IGNORE ACCEPT SHELF
MXX*            RMM   IGNORE ACCEPT SHELF
S0024*          RMM   IGNORE ACCEPT SHELF
S020*           RMM   IGNORE ACCEPT SHELF
THM000:THM029   RMM   IGNORE IGNORE
*               RMM   ACCEPT ACCEPT SHELF
DB*             NORMM IGNORE ACCEPT SHELF
HE00*           NORMM IGNORE ACCEPT SHELF
J5T*            NORMM IGNORE ACCEPT SHELF
MXX*            NORMM IGNORE ACCEPT SHELF
S0024*          NORMM IGNORE ACCEPT SHELF
S020*           NORMM IGNORE ACCEPT SHELF
THM000:THM029   NORMM IGNORE IGNORE
*               NORMM IGNORE IGNORE
```

Figure 8-18 Result of the LISTCONTROL PRTITION

Ensure that you have not specified any definition twice, because in this case the DFSMSrmm initialization process is stopped and message EDG0238E followed by message EDG0215D is written as shown in Figure 8-19.

```
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM71 IN RMM.PARMLIB
EDG0238E OVERLAPPING VOLUME SET DEFINED FOR PRTITION TYPE(ALL). THM*
         OVERLAPS THM*
EDG0215D ERRORS DETECTED IN INITIALIZATION PARAMETERS -
         ENTER ˜Y˜ TO CONTINUE OR ˜N˜ TO CANCEL
```

Figure 8-19 Error message until DFSMSrmm initialization

The following description applies to Figure 8-19:

EDG0238E OVERLAPPING VOLUME SET DEFINED FOR *command* TYPE(*type*). *value1* OVERLAPS *value2*

Explanation A PRTITION or OPENRULE command has specified volume selection operands that conflict with another command.

The following information is included in the message text:

command Is one of the following: PRTITION, or OPENRULE.

type Is one of the following: ALL, RMM, or NORMM.

value1 Is one of the following: a volume serial number, or a volume prefix specified through the VOLUME operand, or a volume range specified with the VOLUMERANGE operand.

value2 Is one of the following: a volume serial number, or a volume prefix specified through the VOLUME operand, or a volume range specified with the VOLUMERANGE operand.

System action DFSMSrmm initialization stops. This message is followed by message “EDG0215D”.

Operator response Notify the system programmer. Reply to message “EDG0215D” as directed. Restart DFSMSrmm when the system programmer has corrected the error.

System programmer response
Correct the VOLUME and VOLUMERANGE operands so that overlaps do not exist. Volume sets must be unique within command and type; for example, PRTITION TYPE(RMM) with VOLUME(A*) and PRTITION TYPE(RMM) with VOLUMERANGE(A:B99999) is an overlap because the volume range A:B99999 begins within the A* set but extends beyond it. When the message includes TYPE(ALL), the overlap might be with another command specifying a type of ALL or it might be with RMM or NORMM. DFSMSrmm ignores the overlapping values if initialization continues.

Source DFSMSrmm

Detecting module EDGSUSE

Routing code 3

Descriptor code 3

8.5.2 Defining OPENRULE commands

Example 8-2 shows the correct use of the EDGRMMnn parmlib member OPENRULE command.

Example 8-2 EDGRMMnn OPENRULE command

```
▶▶──OPENRULE─────VOLUME──────(──Volser_or_prefix──)──────────────────────▶
                 │                                │
                 │─VOLUMERANGE─(──Volser_or_prefix──)─│


      │──TYPE(ALL)───│            │──ANYUSE(ACCEPT)──│
▶──────────────────────────────────────────────────────────────────────◀◀
      │              │            │                  │
      │──Selection───│            │───Intent─────────│


Selection:
│──────────────────────────────────────────────────────────────────────│
      │                      │
      │          │─ALL───│   │
      │──TYPE─(──────────)───│
                 │─RMM───│
                 │─NORMM─│

Intent:
│──────────────────────────────────────────────────────────────────────│
      │                      │
      │            │─ACCEPT─│  │
      │──ANYUSE─(──────────)──│
```

```
|           |-Action-|    |
|                        |
|           |-ACCEPT-|    |
|--INPUT--(------------)--|
|           |-Action-|    |
|                        |
|           |-ACCEPT-|    |
|--OUTPUT-(------------)--|
            |-Action-|
```

Action:

```
|-----------------------------------------------------------------------|
                          |                                   |
                          |-REJECT----------------------------|
                          |          |                    |   |
                          |          |      |----,-------|   |   |
                          |          |      v            |   |   |
                          |          |-BY-(----------------)-|   |
                          |                 |--CATLG------|       |
                          |                 |--SYSID------|       |
                          |                                     |
                          |-IGNORE----------------------------|
                                     |                    |
                                     |      |-ANY--------|   |
                                     |-BY-(----------------)-|
                                            |-NONSPECIFIC-|
                                            |-SPECIFIC----|
```

If there are no OPENRULE or PRTITION commands found in EDGRMMxx parmlib, or there are no REJECT commands, default entries are created from the command shown in Figure 8-20 to cover any volumes for which you do not define an OPENRULE command.

```
OPENRULE VOLUME(*) TYPE(ALL) ANYUSE(ACCEPT)
```

Figure 8-20 Default values of the OPENRULE command

The following descriptions refer to Example 8-2 on page 297:

TYPE(ALL|RMM|NORMM)

Use to identify the type of volumes that are selected by the OPENRULE command. The VOLSER is used to determine whether the volume is defined in the RMM CDS, and, depending on the function, the VOL1 label and HDR1 can be used. Therefore, DFSMSrmm cannot always determine whether the volume is an existing one defined to DFSMSrmm. TYPE is determined in the following manner:

- For system-managed library functions such as import/export, insert/entry, CUA, and eject, only the VOLSER is used to determine whether the volume is defined in the RMM CDS.

- For OPEN processing, the same rules as volume ignore processing are used and the TYPE that is assigned is based on the following information:
  - TYPE(RMM):
    1. The volume serial number is defined in the control data set, and there is no HDR1 tape label for the volume.
    2. The volume serial number is defined in the control data set, and no labels are used (NL, NSL, or BLP with an NL tape).
    3. The 17 characters read from the HDR1 label of the mounted volume match the last 17 characters of the data set name in the control data set.
  - TYPE(NORMM):
    1. The volume serial number is not defined in the control data set.
    2. The volume is defined, but the 17 characters read from the HDR1 label of the mounted volume do not match the last 17 characters of the data set name in the control data set.
- For EXPROC processing, TYPE(RMM) always applies because EXPROC is driven only by volumes defined to DFSMSrmm.

For each set of volumes, you can code one command with RMM and another with NORMM, but only a single command if ALL is used.

RMM The volume must already be defined to DFSMSrmm.

NORMM The volume is not defined to DFSMSrmm.

ALL Applies to all volumes regardless of whether they are defined to DFSMSrmm. ALL is the default value.

VOLUME/VOLUMERANGE

Use these operands to select volumes that are to be managed by this command. You must specify either VOLUME or VOLUMERANGE, and each command defines a set of one or more volumes. Sets cannot overlap within a TYPE, but a set can be a subset of another and such a subset is more specific. When you code a set with TYPE(RMM) and repeat that set with TYPE(NORMM), you can specify different operands for each set. If you want to use the same operands for the sets, you can do so by coding a type of ALL.

VOLUME You can specify the volume as fully qualified or a VOLSER prefix ending in an asterisk (*). A fully qualified volume is one to six alphanumeric, national, or special characters. A VOLSER prefix is zero to five alphanumeric, national, or special characters ending in an asterisk (*). Quotes are required for special characters, and the first character must not be blank. Any value ending in an asterisk (*), even if enclosed in quotes, is considered to be a VOLSER prefix.

VOLUMERANGE Use to select a subset of volumes based on the starting and ending VOLSER. One to six characters for beginning and end of range are required to be processed, and they can be alphanumeric, national, and special characters. Quotes are required for each value regardless of the use of special characters, and the first character must not be blank. The end of range must not be lower than the start of the range.

ANYUSE/INPUT/OUTPUT(action)

Use this operand to identify the type of OPEN request issued by the application. It reflects the application's intent toward the data.

INPUT The application is attempting to read from the tape data set.

OUTPUT The application is attempting to write to the tape data set.

ANYUSE The application is attempting either to read from or write to the tape volume.

ACCEPT/IGNORE/REJECT(by)

Use this operand to identify the action to be taken by DFSMSrmm. The action applies to OPEN processing.

ACCEPT DFSMSrmm processes the volume. This attempt to open a file on the tape volume is also subject to DFSMSrmm open-time volume validation and if allowed, the use of the volume and file is recorded by DFSMSrmm. ACCEPT means that the volume is accepted provided that the validation performed by DFSMSrmm at OPEN time allows the volume to be used.

ACCEPT is the default value.

IGNORE If the requestor is authorized, DFSMSrmm does not process the volume. When the mounted volume matches the requested volume, the use of the volume is ignored by DFSMSrmm; no validation of the volume is performed and there is no recording of the volume or file. The user must be authorized to ignore the volume. DFSMSrmm processing is as if the EDGUX100 exit requested that the volume is ignored. Volume ignore processing is attempted if either IGNORE is specified or the EDGUX100 exit requests ignore. For ignore processing to be successful, the user must be authorized to ignore the volume. The ignore processing is just as if you coded EXPDT=98000 or ACCODE=xCANORES in the JCL and were using the sample EDGUX100 exit shipped with DFSMSrmm.

The use of action IGNORE enables you to ignore any specific or non-specific volume requests, including all those for system-managed volumes.

REJECT DFSMSrmm must prevent OPEN processing from allowing use of the volume. For a specific volume this results in the OPEN request failing. For a non-specific volume, the volume is dismounted by the system and another mount request issued.

BY(SPECIFIC/NONSPECIFIC/ANY)

Use this operand to identify for which requests the IGNORE action applies.

SPECIFIC The IGNORE action only applies if a specific volume is requested. Otherwise, the ACCEPT action is used. NONSPECIFIC means that the IGNORE action only applies if a non-specific request is being processed; the VOLSER in the mount message is either PRIVAT or SCRTCH. Otherwise, the ACCEPT action is used.

ANY The IGNORE action applies to all types of requests.

ANY is the default value.

BY(SYSID, CATLG) Use this operand for volumes of TYPE RMM to specify that the REJECT action only applies, if the volume is defined to DFSMSrmm, the request is for a specific volume, and the BY condition is not met. Otherwise, the ACCEPT action is used.

When specified for TYPE(NORMM), the operand is parsed successfully but is then ignored and never used by DFSMSrmm because volumes of TYPE(NORMM) cannot have existing tape data set records and are not defined to DFSMSrmm.

You can specify one or more of the values SYSID and CATLG. Use this operand to specify that the REJECT action applies if either applied by SYSID or by CATLG. Otherwise, the ACCEPT action is used.

There is no default value.

SYSID Use this operand to specify that for non-scratch volume existing tape data sets, the use of a volume is to be rejected if the creating SYSID of the first file does not match the current SYSID. This operand might help you if you have a common scratch pool across multiple systems but once a volume is used you only want the volume used/referenced on the system from which the data was created. When the creating SYSID matches the current SYSID, the ACCEPT action is used.

CATLG Use this operand to specify that existing tape data sets must be referenced by their catalog entry. When the data set is referenced via its catalog entry, the ACCEPT action is used.

DFSMSrmm checks to see whether there is an existing data set record defined to RMM and applies this rule if the data set record exists regardless of the disposition specified for the data set. In cases where only the first file is being recorded by DFSMSrmm, a data set record does not exist for the second file and the following files, but DFSMSrmm processing assumes that they exist.

Figure 8-21 on page 302 shows the OPENRULE definitions we inserted in our existing EDGRMMnn parmlib member. These rules allow the normal use of all volumes defined to DFSMSrmm “TYPE(RMM)”. All other volumes not defined in the DFSMSrmm control data set can be used if you have the needed access to the STGADMIN.EDG.IGNORE.TAPE.* resource defined in the RACF class FACILITY.

```
/* ************************************************************** */
/*  OPENRULE commands are added at 2008/03/21          Norbert      */
/* ************************************************************** */
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(NORMM)            /* volumes not defined in CDS       */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing            */
                                /*   ignore all types of request    */

OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(RMM)              /* all volumes defined in CDS       */-
         ANYUSE(ACCEPT)         /* allow OPEN processing            */
```

Figure 8-21 EDGRMMnn parmlib OPENRULE definitions

Figure 8-22 shows the correct use and the result of the RMM TSO subcommand LISTCONTROL OPENRULE.

```
RMM LISTCONTROL OPENRULE

Openrule Entries:
                          Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   ACCEPT             ACCEPT
*               NORMM ACCEPT             ACCEPT
```

Figure 8-22 Use and result of the LISTCONTROL OPENRULE

8.5.3 Testing various OPENRULE settings

First, we created a new data set on tape using the JCL as shown in Figure 8-23. In the next test cases, we used this data set for input but we used different OPENRULE settings. The data set was created on volume THS013, which is an IBM 3592 volume inside an IBM automated tape library (ATL).

```
//IKJEFT01 EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RMM.CNTL(OPENR001)
//SYSUT2   DD   DISP=(,KEEP),DSN=MHLRES7.TEST.OPENRULE.RULES,
//         UNIT=ATL3
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-23 Sample JCL used to create a single data set on tape

Our conclusion from the test is that in an unshared tape environment (that is, only a single DFSMSrmm control data set, all tapes are defined in this CDS, and all tape drives are shared), only both OPENRULEs shown in Figure 8-21 and both RACF definitions shown in Figure 8-24 on page 303 are required.

```
RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
        UACC(NONE) OWNER(SYS1    )

RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
        UACC(NONE) OWNER(SYS1    )
```

Figure 8-24 Basic STGADMIN.EDG.TAPE.IGNORE definitions

Test case 1

The following list describes this test case:

Function Read an existing data set on an SMS-managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 8-25 shows the JCL we used.

Result The job ended without any errors, because the user MHLRES7 has access to the data set and the volume can be based on the OPENRULE used on this system.

OPENRULE All DFSMSrmm defined and not defined volumes will be accepted on this system.

RACF The user has access to the data set profile protecting this data set. The user does not need access to the special resources, STGADMIN.EDG.IGNORE.TAPE.RMM.*volser* and STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser* that are defined in the RACF class FACILITY.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-25 Test case 1 sample JCL

The following explanations apply to Figure 8-25:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Uses the IEBGENER utility to read the data set

Figure 8-26 on page 304 shows the result of the RACF SEARCH command that shows that the resource, STGADMIN.EDG.IGNORE.TAPE.RMM.*volser* or STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser*, is not defined at this time.

```
READY
  SR CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

Figure 8-26 Result of test case 1 RACF listing

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 8-27. There is only one OPENRULE definition for each of the RMM and NORMM managed volumes and all tapes are accepted for both definitions.

```
READY
    RMM LC OPE
 Openrule Entries:
                           Input             Output
 Volume or Range Type  Action Condition   Action Condition
 --------------- ----- ------ ----------- ------ -----------
 *               RMM   ACCEPT             ACCEPT
 *               NORMM ACCEPT             ACCEPT

 READY
```

Figure 8-27 Result of test case 1 LISTCONTROL

Figure 8-28 shows the JES message log of this job. There was no error.

```
IEF403I OPENR002 - STARTED - TIME=14.18.28 - ASID=0042 - SC70
-                                         --TIMINGS (MINS.)--
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR002          RACFCMDS    00     28    .00    .00    .00    340   0
-OPENR002          LISTCONT    00     32    .00    .00    .00    347   0
IEF233A M 0B22,THS013,,OPENR002,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
IEF234E K 0B22,THS013,PVT,OPENR002,TESTCASE
-OPENR002          TESTCASE    00     74    .00    .00    .31    217   0
IEF404I OPENR002 - ENDED - TIME=14.18.47 - ASID=0042 - SC70
-OPENR002 ENDED.  NAME-                     TOTAL CPU TIME=   .00  TOTAL
```

Figure 8-28 JES message log of test case 1

Test case 2

The following list describes this test case:

Function Read an existing data set on an SMS-managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 8-29 on page 305 shows the JCL we used.

Result The job abended. Although the user MHLRES7 has access to the data set that resides on this volume, the volume cannot be used on this system. This depends on the OPENRULE definition.

OPENRULE All DFSMSrmm-defined and not defined volumes are ignored on this system.

RACF The user has access to the data set profile that protects this data set. There is an additional need to have READ access to the special resource, STGADMIN.EDG.IGNORE.TAPE.RMM.*, that is defined in the RACF class FACILITY, because the volume is ignored from DFSMSrmm.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-29 Test case 2 sample JCL

The following explanations apply to Figure 8-29:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Uses the IEBGENER utility to read the data set

Figure 8-30 shows the result of the RACF SEARCH command that shows that the resource, STGADMIN.EDG.IGNORE.TAPE.RMM.*volser* or STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser*, is not defined at this time.

```
READ
  SR CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

Figure 8-30 Result of test case 2 RACF listing

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 8-31 on page 306. There is only one OPENRULE definition for RMM and NORMM managed volumes. DFSMSrmm must, depending on this definition, ignore all tape requests on this system.

```
READY
   RMM LC OPE
Openrule Entries:
                           Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   IGNORE ANY         IGNORE ANY
*               NORMM IGNORE ANY         IGNORE ANY
READY
```

Figure 8-31 Result of test case 2 LISTCONTROL

Figure 8-32 shows the JES message log of this job and you can see that the job gets a security violation and an error when opening the tape volume.

```
IEF403I OPENR003 - STARTED - TIME=14.39.26 - ASID=0042 - SC70
-                                          --TIMINGS (MINS.)--
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR003          RACFCMDS    00     28    .00    .00    .00   1213   0
-OPENR003          LISTCONT    00     32    .00    .00    .00    312   0
IEF233A M 0B23,THS013,,OPENR003,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
ICH408I USER(MHLRES7 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)  306
  STGADMIN.EDG.IGNORE.TAPE.RMM.THS013 CL(FACILITY)
  INSUFFICIENT ACCESS AUTHORITY
  FROM STGADMIN.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
EDG4060I VOLUME THS013 REJECTED. OPENRULE ACTION IGNORE BUT USE OUTSIDE
EDG4006E VOLUME THS013 ON 0B23 REJECTED FOR USE BY OPENR003, TESTCASE, S
IEC518I SOFTWARE ERRSTAT: REJTMS   0B23,THS013,SL,OPENR003,TESTCASE
IEC502E R 0B23,THS013,SL,OPENR003,TESTCASE
IEC145I 413-08,IFG0194K,OPENR003,TESTCASE,SYSUT1,0B23,,MHLRES7.TEST.OPEN
IEA848I DUMP SUPPRESSED - USER NOT AUTHORIZED BY SAF
IEF450I OPENR003 TESTCASE - ABEND=S413 U0000 REASON=00000008  313
        TIME=14.39.44
-OPENR003          TESTCASE *S413     40    .00    .00    .30    526   0
IEF404I OPENR003 - ENDED - TIME=14.39.44 - ASID=0042 - SC70
```

Figure 8-32 JES message log of test case 2

The following explanations apply to Figure 8-32:

ICH408I The STGADMIN.** resource is used because there are no more specific RACF resources defined in the RACF class FACILITY. DFSMSrmm uses the normal RACF best matching processing to find a matching profile or resource. The user currently has no access to this definition.

IEC145I The 413-08 is a result of the rejection of the tape volume depending on the OPENRULE definition.

EDG4060I `VOLUME volser REJECTED. OPENRULE ACTION IGNORE BUT USE OUTSIDE OF DFSMSrmm CONTROL NOT AUTHORIZED`

Explanation The volume matched to an OPENRULE entry that specified an action of IGNORE. The IGNORE action requests that DFSMSrmm ignore this volume. The user must be authorized to request that the specified volume is ignored. The user was not authorized and the request failed. Message EDG4060I can be accompanied by an ICH408I message that explains the reason for the authorization failure. If the ICH408I message is not issued, the most likely cause for the error is that there is no security profile defined for ignore processing. When there is no profile, DFSMSrmm does not allow a volume to be ignored. You must add one or more security profiles to ensure the correct authorization.

The following information is included in the message text:

volser The volume serial number of the volume that the user is attempting to use.

System action If DFSMSrmm is operating in warning mode, the volume specified in this message can be used and DFSMSrmm issues message EDG4004I. If DFSMSrmm is operating in protect mode, DFSMSrmm issues message EDG4005E or EDG4006E.

Operator response None.

System programmer response
If the volume needs to be ignored, define one of these RACF resources and authorize the user:
- STGADMIN.EDG.IGNORE.TAPE.*volser*,
- STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser*
- STGADMIN.EDG.IGNORE.TAPE.RMM.*volser*

For information about authorizing users, see the *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Source DFSMSrmm

Detecting module EDGOECM

Routing code 2,3

Descriptor code 3

EDG4006E `VOLUME volser ON rack_number REJECTED FOR USE BY jobname, stepname, ddname; OPEN REQUEST FAILED BY DFSMSrmm`

Explanation Neither the current volume nor any other tape volume can be used for this mount request.

The following information is included in the message text:

volser Volume serial number

rack_number Volume shelf location identifier

job_name Name of a job identified to a system

stepname Name of a step within a job

ddname Data definition name

System action The tape is rejected, and the job abnormally ends.

Operator response None.

System programmer response

Check the specified DD statement for incorrect volume, density, or label parameters. If the DD statement appears correct, review the DFSMSrmm parmlib options to determine whether this was a valid occurrence.

Source: DFSMSrmm

Detecting module: EDGSOCE

Routing code: 2,3,11

Descriptor code: 11

Test case 3

The following list describes this test case:

Function Read an existing data set on an SMS-managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 8-33 shows the JCL we used.

Result The job ended without an error, although the volume must be ignored by the OPENRULE setting. In this case, the user MHLRES7 has READ access to the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* that is now defined in the RACF class FACILITY and has access to the data set residing on the volume.

OPENRULE All DFSMSrmm-defined and not defined volumes are ignored on this system.

RACF The user has access to the data set profile protecting this data set. At a minimum, READ access to the RACF FACILITY class resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is required. The volume must be ignored by DFSMSrmm depending on the OPENRULE TYPE(RMM) definition for DFSMSrmm-managed volumes.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-33 Test case 3 sample JCL

The following explanations apply to Figure 8-33:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Using the IEBGENER utility to read the data set

Figure 8-34 shows the result of the RACF SEARCH command to see that the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is now defined on this system.

```
READ
   RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
          UACC(NONE) OWNER(SYS1    )
     PERMIT STGADMIN.EDG.IGNORE.TAPE.RMM.* -
            ID(MHLRES7) ACC(UPDATE) CLASS(FACILITY)

  SETROPTS GENERIC(FACILITY) REFRESH
  SETROPTS RACLIST(FACILITY) REFRESH
  SR CLASS(FACILITY) MASK(STGADMIN)
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
STGADMIN.EDG.IGNORE.TAPE.RMM.* (G)
READY
```

Figure 8-34 Result of test case 3 RACF listing

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 8-35. You can see that there is only one OPENRULE definition for RMM and NORMM managed volumes, and that for both definitions, all tapes must be ignored.

```
READY
   RMM LC OPE
Openrule Entries:
                          Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   IGNORE ANY         IGNORE ANY
*               NORMM IGNORE ANY         IGNORE ANY

READY
```

Figure 8-35 Result of test case 3 LISTCONTROL

Figure 8-36 shows the JES message log of this job. You can see there was no error because now the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is defined and the user has a minimum of READ access to this resource. If there is a need to write to a volume ignored by DFSMSrmm, the user must have, at a minimum, UPDATE access to this resource.

```
IEF403I OPENR004 - STARTED - TIME=14.49.10 - ASID=0042 - SC70
-                                          --TIMINGS (MINS.)--
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR004          RACFCMDS    00    127    .00    .00    .00   1477   0
-OPENR004          LISTCONT    00     32    .00    .00    .00    309   0
IEF233A M 0B22,THS013,,OPENR004,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
EDG4061I VOLUME THS013 IGNORED. IGNORE REQUESTED BY OPENRULE ACTION IGNO
IEF234E K 0B22,THS013,PVT,OPENR004,TESTCASE
-OPENR004          TESTCASE    00     76    .00    .00    .96    258   0
IEF404I OPENR004 - ENDED - TIME=14.50.09 - ASID=0042 - SC70
```

Figure 8-36 JES message log of test case 3

Test case 4

The following list describes this test case:

Function Read an existing data set on an SMS-managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 8-37 shows the JCL we used.

Result The job ended without an error because the OPENRULE definition is changed to ACCEPT all DFSMSrmm-defined volumes and IGNORE only all volumes not defined in the DFSMSrmm control data set.

OPENRULE All DFSMSrmm-defined tape volumes are accepted and all not defined volumes are ignored on this system.

RACF The user has access to the data set profile protecting this data set. There is no need to have access to the special resources that are defined in the RACF class FACILITY:
STGADMIN.EDG.IGNORE.TAPE.RMM.*
STGADMIN.EDG.IGNORE.TAPE.NORMM.*

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-37 Test case 4 sample JCL

The following explanations apply to Figure 8-37:

RACFCMDS Shows the STGADMIN.EDG.* resources that are defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Uses the IEBGENER utility to read the data set

Figure 8-38 shows the result of the RACF SEARCH command that shows that the resources, STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.NORMM.* are not defined at this time.

```
READ
  SR CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

Figure 8-38 Result of test case 4 RACF listing

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 8-39. You can see that there is an OPENRULE TYPE(RMM) definition for all DFSMSrmm-defined tape volumes with the ACCEPT option that allows the use of volumes defined in the DFSMSrmm control data set. All other volumes that are not defined in the DFSMSrmm control data set can only be used when the user has access to the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser* that is defined in the RACF class FACILITY.

```
READY
   RMM LC OPE
Openrule Entries:
                          Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   ACCEPT             ACCEPT
*               NORMM IGNORE ANY         IGNORE ANY

READY
```

Figure 8-39 Result of test case 4 LISTCONTROL

In Figure 8-40, we show the JES message log of this job. You can see that there was no error.

```
IEF403I OPENR005 - STARTED - TIME=15.31.12 - ASID=0042 - SC70
-                                          --TIMINGS (MINS.)--
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR005          RACFCMDS    00     71    .00    .00    .00   1192   0
-OPENR005          LISTCONT    00     32    .00    .00    .00    315   0
IEF233A M 0B22,THS013,,OPENR005,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
IEF234E K 0B22,THS013,PVT,OPENR005,TESTCASE
-OPENR005          TESTCASE    00     74    .00    .00    .29    230   0
IEF404I OPENR005 - ENDED - TIME=15.31.30 - ASID=0042 - SC70
-OPENR005 ENDED.  NAME-                      TOTAL CPU TIME=   .00  TOTAL
```

Figure 8-40 JES message log of test case 4

Test case 5

The following list describes this test case:

Function In the DFSMSrmm control data set, we change the data set name DFSMSrmm has automatically recorded at Open/Close/End of Volume (O/C/EOV) time for tape volume THS013 to simulate a duplicate tape volume or a *foreign* tape volume, which is a tape that comes from outside. The data set name that is used in the JCL no longer matches the data set name DFSMSrmm has recorded for this volume. To read this data set, we also used IEBGENER. The used volume serial number is THS013. Figure 8-41 on page 312 shows the JCL we used.

Result The job abended, although the user MHLRES7 has access to the data set. The volume cannot be used on this system because of the OPENRULE TYPE(NORMM) definition. This rule is used by DFSMSrmm, because the data set name that is used in our JCL does not match the data set name DFSMSrmm recorded for this volume.

OPENRULE All DFSMSrmm-defined volumes are accepted by the OPENRULE TYPE(RMM) ACCEPT and can be used under normal conditions but

all volumes that are not defined in the DFSMSrmm control data set are ignored on this system, depending on the OPENRULE TYPE(NORMM) IGNORE definition.

RACF The user has access to the data set profile protecting this data set. There is a requirement to have READ access to the special resource STGADMIN.EDG.IGNORE.TAPE.NORMM.* that is defined in the RACF class FACILITY, because the volume is ignored by DFSMSrmm based on the OPENRULE definition TYPE(NORMM), but this resource is not defined at this time.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-41 Test case 5 sample JCL

The following explanations apply to Figure 8-41:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Uses the IEBGENER utility to read the data set

Figure 8-42 shows the result of the RACF SEARCH command that shows that the resources, STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.NORMM.*, are not defined at this time. The resource STGADMIN.EDG.FORCE is now present because we have a need to change the Open/Close/End of Volume (O/C/EOV) recorded data set name for this volume using the RMM CHANGEVOLUME subcommand with the FORCE option.

```
READ
  SR CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.FORCE
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

Figure 8-42 Result of test case 5 RACF listing

The result of the RMM TSO LISTCONTROL and SEARCHDATASET subcommands is shown in Figure 8-43 on page 313. You can see that there is an OPENRULE definition TYPE(NORMM) for all volumes that are not managed with DFSMSrmm and that definition has an action of IGNORE for any request to these volumes. Also, you can see that the data set name that is recorded for this volume in DFSMSrmm does not match the data set name

we are using in our JCL. This volume is a foreign volume to DFSMSrmm, so the OPENRULE with TYPE(NORMM) is used.

```
READY
   RMM LC OPE
Openrule Entries:
                           Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   ACCEPT             ACCEPT
*               NORMM IGNORE ANY         IGNORE ANY

READY

   RMM SD VOLUME(THS013)
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- -----
MHLRES7.TEST.FROM.EXTERNAL                   THS013 MHLRES7  2008/081    1
1 ENTRY LISTED
```

Figure 8-43 Result of test case 5 LISTCONTROL and SEARCHDATASET

In Figure 8-44 on page 314, we show you the JES message log of this job, and you can see the job abended and cannot open the volume depending on the OPENRULE definition. Also, you can see the missing access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser* resource defined in the RACF class FACILITY.

```
IEF403I OPENR006 - STARTED - TIME=16.03.47 - ASID=0042 - SC70
-                                          --TIMINGS (MINS.)--
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR006          RACFCMDS    00     66    .00    .00    .00   1196   0
-OPENR006          LISTCONT    00     32    .00    .00    .00    314   0
IEF233A M 0B23,THS013,,OPENR006,TESTCASE,MHLRES1.TEST.OPENRULE.RULES
ICH408I USER(MHLRES7 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)  847
  STGADMIN.EDG.IGNORE.TAPE.NORMM.THS013 CL(FACILITY)
  INSUFFICIENT ACCESS AUTHORITY
  FROM STGADMIN.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
EDG4060I VOLUME THS013 REJECTED. OPENRULE ACTION IGNORE BUT USE OUTSIDE
EDG4006E VOLUME THS013 ON 0B23 REJECTED FOR USE BY OPENR006, TESTCASE, S
IEC518I SOFTWARE ERRSTAT: REJTMS   0B23,THS013,SL,OPENR006,TESTCASE
IEC502E R 0B23,THS013,SL,OPENR006,TESTCASE
IEC145I 413-08,IFG0194K,OPENR006,TESTCASE,SYSUT1,0B23,,MHLRES1.TEST.OPEN
IEA848I DUMP SUPPRESSED - USER NOT AUTHORIZED BY SAF
IEF450I OPENR006 TESTCASE - ABEND=S413 U0000 REASON=00000008  854
        TIME=16.04.05
-OPENR006          TESTCASE *S413     38    .00    .00    .29    782   0
IEF404I OPENR006 - ENDED - TIME=16.04.05 - ASID=0042 - SC70
```

Figure 8-44 JES message log of test case 5

Note: The STGADMIN.** resource is used because there are no more specific RACF resources defined in the RACF class FACILITY. DFSMSrmm uses the normal RACF best matching processing to find a matching profile or resource. The user currently has no access to this definition.

"Test case 2" on page 304 shows the descriptions of the EDGnnnnc messages.

Test case 6

The following list describes this test case:

Function In the DFSMSrmm control data set, we change the data set name DFSMSrmm has automatically recorded at Open/Close/End of Volume (O/C/EOV) time for tape volume THS013 to simulate a duplicate tape volume or a foreign tape volume, which is a tape that comes from outside. The data set name that is used in the JCL no longer matches the data set name DFSMSrmm has recorded for this volume. To read this data set, we also used IEBGENER. The volume serial number is THS013. Figure 8-45 on page 315 shows the JCL we used.

Result The job ended without an error, because the user MHLRES7 has access to the data set and has the required access to the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.*, depending on the OPENRULE TYPE(NORMM) IGNORE definition used for all volumes that are not defined in the DFSMSrmm control data set.

OPENRULE All DFSMSrmm-defined volumes are accepted by the OPENRULE TYPE(RMM) ACCEPT and can be used under normal conditions, but all volumes that are not defined in the DFSMSrmm control data set are

ignored on this system depending on the OPENRULE TYPE(NORMM) IGNORE definition.

RACF The user has access to the data set profile protecting this data set and also has access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.* special resource that is defined in the RACF class FACILITY, because the volume is ignored by DFSMSrmm based on the OPENRULE TYPE(NORMM) definition.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.TEST.OPENRULE.RULES,
//         UNIT=ATL3,VOL=SER=THS013
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
```

Figure 8-45 Test case 6 sample JCL

The following explanations apply to Figure 8-45:

RACFCMDS Shows the STGADMIN.EDG.* resources that are defined in the RACF class FACILITY

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command

TESTCASE Uses the IEBGENER utility to read the data set

Figure 8-46 on page 316 shows the result of the RACF commands we used to show you that we deleted and newly added the STGADMIN.EDG.IGNORE.TAPE.NORMM.* resource in the RACF class FACILITY. This resource is used if a volume is not defined in the DFSMSrmm control data set. The resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is used if a volume is defined in the DFSMSrmm control data set.

The user MHLRES7 has UPDATE access to the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.*, but only READ access is required to read a volume. It is normal RACF processing, which means that UPDATE access has more authority than READ access and solves the request. We also defined the resource STGADMIN.EDG.IGNORE.TAPE.RMM.*, but this resource is not used at this time.

```
READY
   RDELETE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.*
READY
   RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
           UACC(NONE) OWNER(SYS1    )
READY
      PERMIT STGADMIN.EDG.IGNORE.TAPE.RMM.* -
             ID(MHLRES7) ACC(UPDATE) CLASS(FACILITY)
READY

   RDELETE FACILITY STGADMIN.EDG.IGNORE.TAPE.NORMM.*
READY
   RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
           UACC(NONE) OWNER(SYS1    )
READY
      PERMIT STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
             ID(MHLRES7) ACC(UPDATE) CLASS(FACILITY)
READY

   SETROPTS GENERIC(FACILITY) REFRESH
READY
   SETROPTS RACLIST(FACILITY) REFRESH
READY
   SR CLASS(FACILITY) MASK(STGADMIN)
READY
  STGADMIN.EDG.FORCE
  STGADMIN.EDG.LISTCONTROL
  STGADMIN.EDG.MASTER
  STGADMIN.EDG.IGNORE.TAPE.NORMM.* (G)
  STGADMIN.EDG.IGNORE.TAPE.RMM.* (G)
```

Figure 8-46 Result of test case 6 RACF listing

The result of the RMM TSO LISTCONTROL and SEARCHDATASET subcommands is shown in Figure 8-47 on page 317. You can see that there is an OPENRULE TYPE(NORMM) definition for all non-DFSMSrmm-managed volumes and that definition has an action of IGNORE. Any request to this volume must be ignored by DFSMSrmm. Also, you can see that the data set name for this volume, which is recorded in DFSMSrmm, did not match the data set name we are using in our JCL. This volume is a foreign volume to DFSMSrmm, so the OPENRULE TYPE(NORMM) is used.

```
READY
   RMM LC OPE
Openrule Entries:
                          Input             Output
Volume or Range Type  Action Condition   Action Condition
--------------- ----- ------ ----------- ------ -----------
*               RMM   ACCEPT             ACCEPT
*               NORMM IGNORE ANY         IGNORE ANY

READY

   RMM SD VOLUME(THS013)
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- -----
MHLRES7.TEST.FROM.EXTERNAL                   THS013 MHLRES7  2008/081    1
1 ENTRY LISTED
```

Figure 8-47 Result of test case 6 LISTCONTROL and SEARCHDATASET

In Figure 8-48, we show you the JES message log of this job. The job ended without an error, depending on the correct access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.* resource defined in the RACF class FACILITY.

```
IEF403I OPENR007 - STARTED - TIME=16.12.55 - ASID=004B - SC70
-                                         --TIMINGS (MINS.)--            --
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
-OPENR007          RACFCMDS    00    164    .00    .00    .00   1854   0
-OPENR007          LISTCONT    00     32    .00    .00    .00    312   0
IEF233A M 0B22,THS013,,OPENR007,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
EDG4061I VOLUME THS013 IGNORED. IGNORE REQUESTED BY OPENRULE ACTION IGNORE
IEF234E K 0B22,THS013,PVT,OPENR007,TESTCASE
-OPENR007          TESTCASE    00     75    .00    .00    .30    244   0
IEF404I OPENR007 - ENDED - TIME=16.13.14 - ASID=004B - SC70
-OPENR007 ENDED.  NAME-                     TOTAL CPU TIME=   .00  TOTAL EL
```

Figure 8-48 JES message log of test case 6

The following explanations apply to Figure 8-48:

EDG4061I VOLUME *volser* IGNORED. IGNORE REQUESTED BY OPENRULE ACTION IGNORE

Explanation The DFSMSrmm parmlib options for OPENRULE with an action of IGNORE requested that this volume will be ignored by DFSMSrmm. DFSMSrmm does not record any information about the specified volume and permits the volume's use because the user is authorized to use a volume that is ignored or DFSMSrmm is running in record mode.

The following information is included in the message text:

volser The volume serial number of the volume that the user is attempting to use

System action	DFSMSrmm ignores this volume while it remains mounted. DFSMSrmm does not validate the mounted volume and does not record any information about the current tape usage in the DFSMSrmm control data set.
Operator response	None
Source	DFSMSrmm
Detecting module	EDGOECM
Routing code	2,3
Descriptor code	3

8.6 Converting REJECT commands

If you use REJECT commands, you must convert from the use of REJECT commands in order to use the PRTITION and OPENRULE commands. Review your existing REJECT commands across all system images and identify the sets of volumes to be used on each system. Identify the PRTITION and OPENRULE commands that are required for each system to minimize parmlib maintenance as your storage requirements change. Until you define one or more PRTITION or OPENRULE commands, both partitioning and rejecting volumes are controlled by REJECT commands. Any REJECT commands that specify ANYUSE are used for partitioning of undefined volumes, but all REJECT commands are used for rejecting volumes at OPEN time.

When you use either PRTITION or OPENRULE commands, the REJECT commands are no longer used so you must start using both PRTITION and OPENRULE at the same time to avoid any loss of function. You must remove the REJECT commands from parmlib, because they will fail when any PRTITION or OPENRULE commands are defined. When parmlib is processed, DFSMSrmm issues message EDG0239E, followed by message EDG0215D to provide an option to ignore the error or fail and then retry with message EDG0107D with another parmlib member.

When you have created your new commands, remove the REJECT commands. The PRTITION and OPENRULE commands can only be used on z/OS V1R10 and later releases. Lower-level releases continue to use the REJECT commands. To give you more choice and flexibility, do not convert from REJECT commands one-to-one and do not automate the conversion. When each REJECT command is converted strictly to an equivalent PRTITION and OPENRULE command, you can end up with too much complexity and duplication. The best approach is to start from scratch and list the basic rules that you want to implement. For example, consider the changed function and whether you are affected by it:

- For volumes that are not defined to DFSMSrmm, the REJECT command with ANYUSE causes CUA requests to fail. With the PRTITION command, you can choose between the actions of either ACCEPT or IGNORE. You no longer have the option to fail a CUA request. The ACCEPT action results in the volume being added to DFSMSrmm and the CUA continues. The IGNORE action causes DFSMSrmm to take no action for the CUA request.
- For non-system-managed volumes that are not defined to DFSMSrmm, the default processing during OPEN did not add the volume to the CDS. The default processing for the PRTITION command is that non-system-managed volumes are added to the CDS.

8.6.1 Examples of converting REJECT commands

This section provides examples that show the conversion of REJECT commands to OPENRULE and PRTITION commands.

The sample commands in Figure 8-49 show REJECT used by both partitioning and for open time rejects. A REJECT with a prefix, when used at open time, applies only to volumes defined to RMM so TYPE(RMM) is used for OPENRULE. However, partitioning REJECT ANYUSE applies only to volumes *not* defined to RMM, so TYPE(NORMM) is used for PRTITION.

```
REJECT ANYUSE(prefix)

OPENRULE VOLUME(prefix) -
         TYPE(RMM) -
         ANYUSE(REJECT)

PRTITION VOLUME(prefix) -
         TYPE(NORMM) -
         SMT(IGNORE) -
         NOSMT(IGNORE)
```

Figure 8-49 Conversion of REJECT ANYUSE

The sample commands in Figure 8-50 show that REJECT is used only for open-time rejects. A REJECT with a prefix, when used at open time, applies only to volumes defined to RMM so TYPE(RMM) is used for OPENRULE. The REJECT OUTPUT allows input processing, so code the INPUT option on the OPENRULE for completeness, although ACCEPT is the default.

```
REJECT OUTPUT(prefix)

OPENRULE VOLUME(prefix) -
         TYPE(RMM) -
         OUTPUT(REJECT) -
         INPUT(ACCEPT)
```

Figure 8-50 Conversion of REJECT OUTPUT

The sample commands in Figure 8-51 on page 320 show REJECT is used by both partitioning and for open-time rejects. A REJECT with PREFIX(*) applies only to volumes that are *not* defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

```
REJECT ANYUSE(*)

OPENRULE VOLUME(*) -
         TYPE(NORMM) -
         ANYUSE(REJECT)

PRTITION VOLUME(*) -
         TYPE(NORMM) -
         SMT(IGNORE) -
         NOSMT(IGNORE)
```

Figure 8-51 Conversion of REJECT ANYUSE(*)

The sample commands in Figure 8-52 on page 321show the following information:

- Three systems, SYSA, SYSB, and SYSC, are each allowed to use only a single range of system-managed volumes (JT, JP, and JX).
- Each system has its own CDS and TCDB.
- System-managed volumes are not automatically defined to DFSMSrmm.
- Private volumes can be shared.
- Undefined non-system-managed volumes can be used for input.
- EDGUX100 is customized to automate ignoring private system-managed volumes from other partitions and ignoring undefined non-system-managed volumes. This is based on the RMM API checking whether a volume is defined to RMM. The following RACF FACILITY class profiles are also required:
  - STGADMIN.EDG.IGNORE.TAPE.NORMM.JP*
  - STGADMIN.EDG.IGNORE.TAPE.NORMM.JX*

```
/* SYSA example */
REJECT ANYUSE(JP*)
REJECT ANYUSE(JX*)
REJECT ANYUSE(*)

/* Allow read of undefined volumes */
OPENRULE VOLUME(*) -
         TYPE(NORMM) -
         OUTPUT(REJECT)

/* Ignore for input selected system managed volumes if authorized */
OPENRULE VOLUME(JP*) -
         TYPE(NORMM) -
         INPUT(IGNORE) -
         OUTPUT(REJECT)

OPENRULE VOLUME(JX*) -
         TYPE(NORMM) -
         INPUT(IGNORE) -
         OUTPUT(REJECT)

/* Global partition rule - ignore all smt volumes */
PRTITION VOLUME(*) -
         TYPE(RMM) -
         SMT(IGNORE) -
         NOSMT(ACCEPT)

PRTITION VOLUME(*) -
         TYPE(NORMM) -
         SMT(IGNORE) NOSMT(IGNORE)

/* This partition owns JT* volumes if predefined */
PRTITION VOLUME(JT*) -
         TYPE(RMM) -
         SMT(ACCEPT)
```

Figure 8-52 Conversion of REJECT ANYUSE for three systems

8.7 Conversion from REJECT to PRTITION and OPENRULE

If you want to implement the OPENRULE and PRTITION commands but are already using REJECT commands in parmlib, you must plan for conversion.

Until you define one or more PRTITION or OPENRULE commands, both partitioning and rejecting volumes are controlled by REJECT commands. Any REJECT commands that specify ANYUSE are used for partitioning of undefined volumes, but all REJECT commands are used for rejecting volumes at OPEN time.

When you use either PRTITION or OPENRULE commands, the REJECT commands are no longer used so you must plan to start using both PRTITION and OPENRULE at the same time to avoid loss of function.

You need to remove the REJECT commands from parmlib because they will fail when any PRTITION or OPENRULE commands are defined. When parmlib is processed, DFSMSrmm issues EDG0239E and then EDG0215D to provide an option to ignore the error or fail and then retry via EDG0107D with another parmlib member. When you have created your new commands, remove the REJECT commands.

The PRTITION and OPENRULE commands can only be used on z/OS V1R10 and later releases. Lower-level releases continue to use the REJECT commands.

Conversion from REJECT commands is not best if done one-to-one; therefore, do not automate this conversion. When each REJECT command is converted strictly to an equivalent PRTITION and OPENRULE command, you can end up with too much complexity and also duplication. The best approach is to start from scratch and list the basic rules you want to implement, for example:

- For partitioning, all non-defined volumes are ignored.
- For open rules, all volumes that are defined to DFSMSrmm can be used for both input and output.

When you have the basic rules, you can next identify sets of volumes and how each is to be treated and managed. For example, the only volumes to be used on a system/partition might be all those volumes with the ‘A’ prefix. You can create a command that uses either VOLUME(A*) or VOLUMERANGE(‘A’:’A99999’) and select the TYPE based on whether you will always predefine the volumes or allow them to be defined during cartridge entry and OPEN processing.

For the PRTITION commands, you must consider the system-managed and non-system-managed volumes separately. Be careful not to specify TYPE(RMM/ALL) with NOSMT(IGNORE) unless you really want EXPROC processing to skip the volumes. However, TYPE(RMM/ALL) with SMT(IGNORE) makes sense because DFSMSrmm causes OAM to ignore the volume and leave it for another system or partition, and it also causes DFSMSrmm to skip EXPROC without checking whether the volume is in the TCDB.

Consider the new function provided with the PRTITION and OPENRULE commands and whether it makes sense to exploit it:

- You can have separate commands based on whether the volume is defined to DFSMSrmm.
- OPENRULE provides an automated way to ignore volumes at open time without the need to customize.
- EDGUX100 or use EXPDT=98000 in JCL. OPENRULE also allows you to control the use of existing data sets, such as enforcing cataloging and reference only from the creating system.

Consider the changed function and whether you are affected by it:

- For volumes not defined to DFSMSrmm, the REJECT command with ANYUSE caused CUA requests to fail. With the new PRTITION command, you can choose between the actions of either ACCEPT or IGNORE. You no longer have the option to fail a CUA request. The ACCEPT action results in adding the volume to DFSMSrmm and the CUA continues. The IGNORE action causes DFSMSrmm to take no action for the CUA request.
- For non-system-managed volumes, which are not defined to DFSMSrmm, the default processing during OPEN was to not add the volume to the CDS. The default processing for PRTITION is that non-system-managed volumes are added to the CDS.

Look at your use of the EDGUX100, EDGUX200, and CBRUXENT installation exits to determine whether they have been customized and might no longer be needed. The PRTITION and OPENRULE function allows you to remove almost any customization related to partitioning, rejecting volumes, and ignoring volumes.

Your installation exit customization does not need to be removed immediately but it can be run with the new function, and only removed when you are satisfied that it is no longer needed.

Any existing customization of the CBRUXENT and EDGUX200 exits can likely be removed and replaced with the use of PRTITION parmlib commands. For EDGUX100, any existing function that controls the use of a volume, that might fail open or reject a volume, or enable volumes to be ignored without coding EXPDT=98000, can likely be removed and replaced by the use of the OPENRULE parmlib commands.

8.7.1 Partitioned library with shared CDS and TCDB

The first example in Figure 8-53 shows a system-managed tape library partitioned by the following methods:

- Using the CBRUXENT user exits on each system
- Using the EDGUX200 user exit on each system
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system, but all volumes in the PRIVATE category are shared between the two systems.

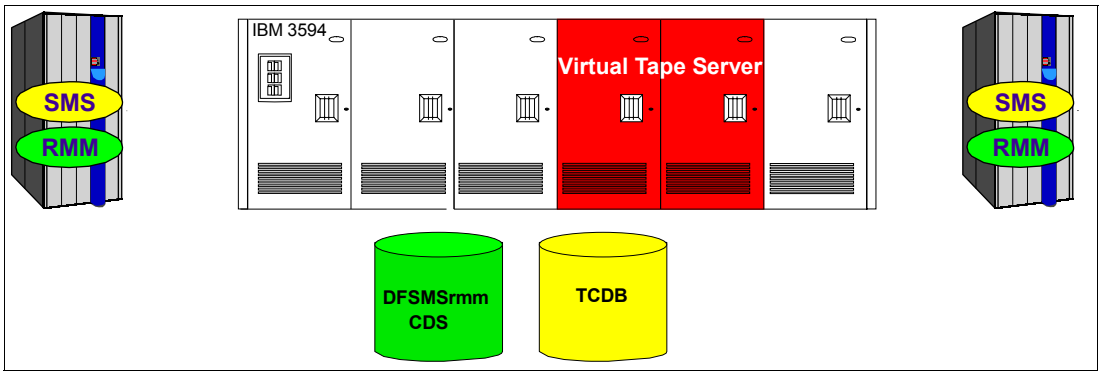


Figure 8-53 Partitioned library with shared CDS and TCDB

In this case, all volumes are shared between the two systems and can be used on both systems. All volumes that are not defined to the DFSMSrmm control data sets are ignored.

Figure 8-54 on page 324 shows the new OPENRULE commands that must be defined in the EDGRMMnn parmlib member to replace the CBRUXENT and EDGUX200 user exit modifications.

```
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(NORMM)            /* volumes not defined in CDS       */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing            */
                                /*   ignore all types of request    */

OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(RMM)              /* all volumes defined in CDS       */-
         ANYUSE(ACCEPT)         /* allow OPEN processing            */
```

Figure 8-54 Convert a single CDS and TCDB to OPENRULE and PRTITION commands

8.7.2 Partitioned library with shared TCDB and different CDSs

Figure 8-55 shows a system-managed tape library partitioned by the following methods:

- Different DFSMSrmm control data set for each system
- Use of REJECT ANYUSE in EDGRMMnn parmlib member
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the CDS, but all volumes in the PRIVATE category are shared between the two systems. To access these volumes, specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 user exit is needed.

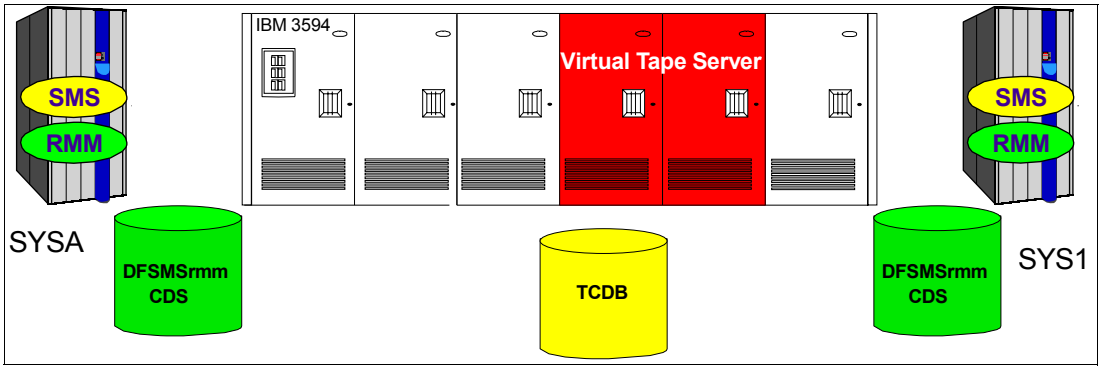


Figure 8-55 Partitioned library with shared TCDB and different CDSs

In this case, we have two systems sharing a DFSMSrmm control data set, but each system has a separate tape control database (TCDB). On system SYSA, we are using only volumes starting with a volume prefix A and on system SYS1 we are using volumes starting with the prefixes 1* and 2*. In both cases, the use of volumes from the other system is not allowed.

Figure 8-56 on page 325 shows the new PRTITION and OPENRULE commands that must be defined in the EDGRMMnn parmlib member to replace the EDGUX100 user exit. The conversion to this command allows you to define new volumes in the library manager database. DFSMSrmm accepts the addition of these volumes in the DFSMSrmm control data set if the prefix matches the prefix we specified in the PRTITION command. You can also add the volumes first in the DFSMSrmm control data set and by using the add volume command on the library manager or inserting the volume into the library later.

On system SYSA, all tapes with a prefix A* can be used inside and outside the ATL, but on system SYS1 the use of tape volumes outside the ATL is not allowed.

```
global setting:
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(NORMM)            /* volumes not defined in CDS       */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing            */
                                /*   ignore all types of request    */

OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(RMM)              /* all volumes defined in CDS       */-
         ANYUSE(ACCEPT)         /* allow OPEN processing            */

PRTITION VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(NORMM)            /* not defined in DFSMSrmm          */-
         SMT(IGNORE)            /* ignore SMS managed volumes       */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */


SYSA:
PRTITION VOLUME(A*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(IGNORE)            /* ignore SMS managed volumes       */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */

SYS1:
OPENRULE VOLUME(A*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         INPUT(ACCEPT)          /* allow OPEN input processing      */

PRTITION VOLUME(1*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(ACCEPT)            /* process the volume               */-
         NOSMT(ACCEPT)          /* process the volume               */

PRTITION VOLUME(2*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(ACCEPT)            /* process the volume               */-
         NOSMT(ACCEPT)          /* process the volume               */
```

Figure 8-56 Convert a single TCDB with two CDSs to OPENRULE and PRTITION commands

8.7.3 Partitioned library with unshared CDS and TCDB

In the third example, shown in Figure 8-57 on page 326, you can see a system-managed tape library partitioned by these methods:

- By system
- Use of REJECT ANYUSE in EDGRMMnn parmlib member
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system. All volumes are not shared, so define a volume in private status in the TCDB first. To access this volume, specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 user exit is needed.

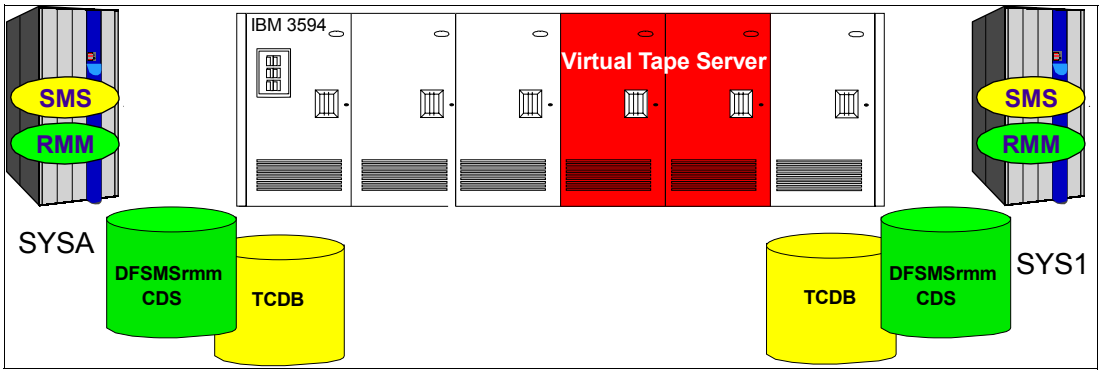


Figure 8-57 Partitioned library with unshared CDS and TCDB

This is a case where REJECT is used for both partitioning and for open-time rejects. A REJECT with PREFIX(*) applies only to volumes that are *not* defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

Figure 8-58 shows the new PRTITION and OPENRULE commands that must be defined in the EDGRMMnn parmlib member to replace the EDGUX100 user exit.

```
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS           */-
         TYPE(NORMM)            /* volumes not defined in CDS        */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing             */
                                /*   ignore all types of request     */

OPENRULE VOLUME(*)              /*     added 2008/03/21 NS           */-
         TYPE(RMM)              /* all volumes defined in CDS        */-
         ANYUSE(ACCEPT)         /* allow OPEN processing             */

PRTITION VOLUME(*)              /*     added 2008/03/21 NS           */-
         TYPE(NORMM)            /* not defined in DFSMSrmm           */-
         SMT(IGNORE)            /* ignore SMS managed volumes        */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */

SYSA:
PRTITION VOLUME(A*)             /*     added 2008/03/21 NS           */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(ACCEPT)            /* process the volume                */-
         NOSMT(ACCEPT)          /* process the volume                */

SYS1:
PRTITION VOLUME(1*)             /*     added 2008/03/21 NS           */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(ACCEPT)            /* process the volume                */-
         NOSMT(ACCEPT)          /* process the volume                */
```

Figure 8-58 Convert two systems sharing a library to OPENRULE and PRTITION commands

8.7.4 Partitioned library with shared CDS and unshared TCDB

In the last example in Figure 8-11 on page 289, we showed a system-managed library partitioned by these methods:

- By system
- By using a customized version of CBRUXENT user exits on each system
- By different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is run by the system based on the TCDB. All volumes are not shared so you must define a volume in private status in the TCDB before you can use it. Figure 8-9 on page 288 shows how you can add a volume in the TCDB using AMS commands. After you successfully process the volume that you previously defined in the TCDB, delete the entry from the TCDB using the command as shown in Figure 8-59.

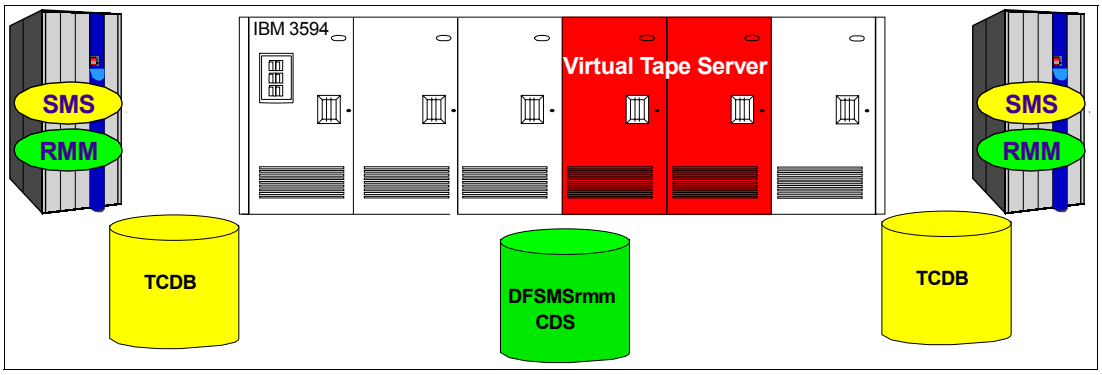


Figure 8-59 Partitioned library with shared CDS and unshared TCDB

This is a case where REJECT is used by both partitioning and for open-time rejects. A REJECT with PREFIX(*) applies only to volumes that are *not* defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

Figure 8-60 on page 328 shows the new PRTITION and OPENRULE commands that must be defined in the EDGRMMnn parmlib member to replace the customized CBRUXENT user exit. The conversion to these commands allows you to define new volumes in the library manager database. DFSMSrmm accepts the addition of these volumes in the DFSMSrmm control data set if the prefix matches the prefix we specified in the PRTITION command. You can also add the volumes first in the DFSMSrmm control data set and use the add volume command on the library manager or insert the volume into the library later.

```
OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(NORMM)            /* volumes not defined in CDS       */-
         ANYUSE(IGNORE BY(ANY)) /* bypass RMM processing            */
                                /*   ignore all types of request    */

OPENRULE VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(RMM)              /* all volumes defined in CDS       */-
         ANYUSE(ACCEPT)         /* allow OPEN processing            */

PRTITION VOLUME(*)              /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(IGNORE)            /* ignore SMS managed volumes       */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */

SYSA:
PRTITION VOLUME(A*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(IGNORE)            /* ignore SMS managed volumes       */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */

SYS1:
PRTITION VOLUME(1*)             /*     added 2008/03/21 NS          */-
         TYPE(ALL)              /* volumes defined/not defined in CDS*/-
         SMT(IGNORE)            /* ignore SMS managed volumes       */-
         NOSMT(IGNORE)          /* ignore volumes outside an ATL/VTS */
```

Figure 8-60 Convert two TCDBs with one CDS to OPENRULE and PRTITION commands

9

Catalog synchronization

This chapter explains how to synchronize your MVS user catalogs with Data Facility System Managed Storage removable media manager (DFSMSrmm) to reduce catalog locates.

This chapter includes the following information:

- Overview of catalog synchronization
- Preparing DFSMSrmm catalog synchronization
- Synchronizing with a shared catalog environment
- Synchronizing when catalogs are not shared

This chapter also provides detailed information about the implementation of the DFSMSrmm catalog synchronization support.

9.1 Overview of catalog synchronization

When DFSMSrmm is installed on all systems that share the control data set (CDS), DFSMSrmm dynamically records all catalog updates for tape data sets in the DFSMSrmm CDS. DFSMSrmm always tracks tape data set catalog activity independently of the CATSYSID parameter.

When you enable catalog synchronization, DFSMSrmm synchronizes the DFSMSrmm CDS and available user catalogs. Enabling catalog synchronization is normally a one-time task. When the catalog status is synchronized, DFSMSrmm continually tracks and updates the catalog status as shown in Figure 9-1. In this figure, CAS means *catalog address space*.

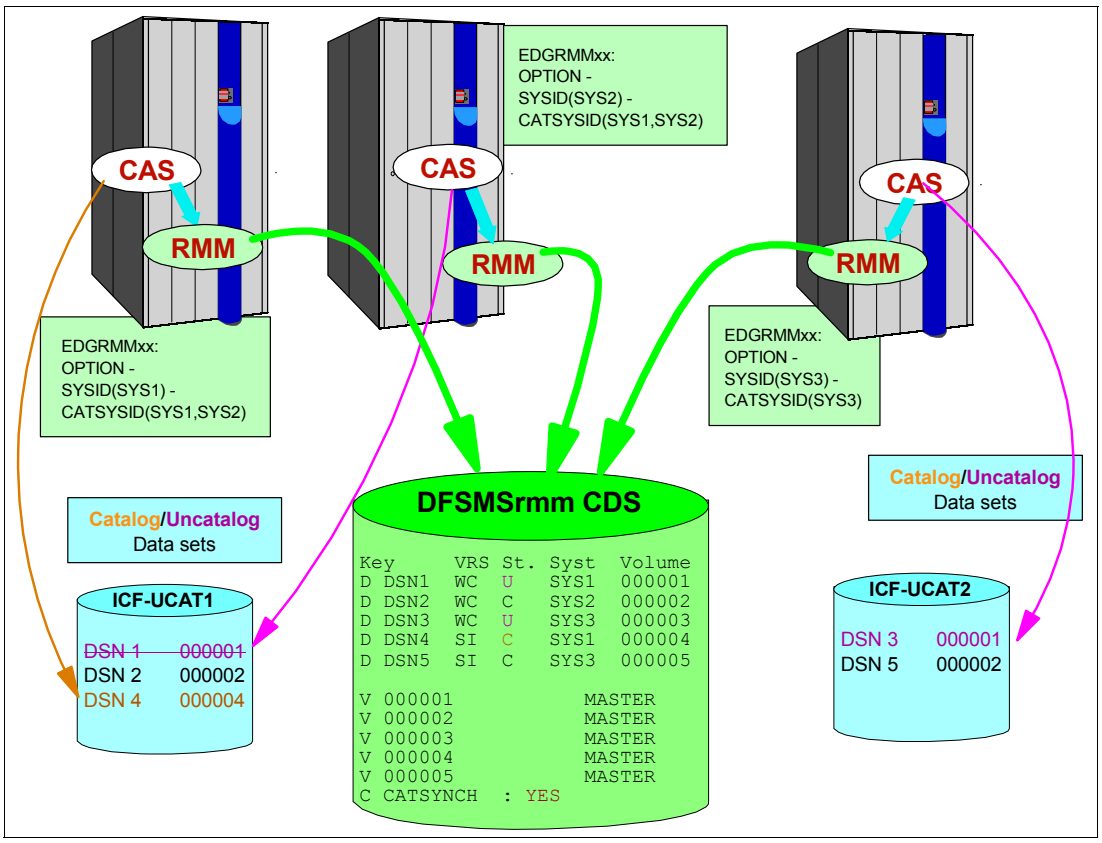


Figure 9-1 Catalog status tracking

You enable catalog synchronization by specifying the DFSMSrmm EDGRMMxx OPTION CATSYSID operand. When you specify the CATSYSID operand, DFSMSrmm uses the catalog search interface to obtain catalog information used for DFSMSrmm vital record processing.

DFSMSrmm performs checking to determine that the catalog search interface catalog environment does not have any errors that might prevent successful processing. DFSMSrmm issues messages EDG2235E, EDG2236I, or EDG2237E when an error is encountered. If you do not specify the DFSMSrmm EDGRMMxx OPTION CATSYSID operand, DFSMSrmm vital record processing uses catalog locates to determine whether data sets are still cataloged.

For detailed information about catalog synchronization, see the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

9.1.1 DFSMSrmm catalog processing

When you run DFSMSrmm with user catalogs and the DFSMSrmm CDS unsynchronized, DFSMSrmm issues catalog locates as required to check whether data sets are cataloged. Catalog locates use the standard catalog search to determine whether a data set is cataloged. Any data sets that are cataloged using JOBCAT or STEPCAT might not be found to be cataloged because a standard catalog search might not find them.

When you run DFSMSrmm with user catalogs and the DFSMSrmm CDS synchronized, DFSMSrmm does not need to issue catalog locates to find out whether a data set is cataloged. The catalog status tracked by DFSMSrmm in the CDS is used to determine if a data set is cataloged. When DFSMSrmm tracks catalog status, it does so regardless of whether JOBCAT or STEPCAT is used.

During catalog synchronization, DFSMSrmm uses the Catalog Search Interface (CSI) to retrieve data set catalog information. CSI returns catalog information for all data sets in all catalogs that are in or connected to the master catalog. Because of this, DFSMSrmm can detect that a data set is cataloged even if it cannot be found using the standard catalog search.

9.1.2 Reasons to re-synchronize catalog synchronization

Set up automatic synchronization by automating the responses to DFSMSrmm messages EDG8200E and EDG8201E. Run the EDGUTIL UPDATE with the SYSIN command CONTROL CDSID(PROD) CATSYNCH(NO). You receive these error messages when there is an error in DFSMSrmm catalog processing.

You must re-synchronize the DFSMSrmm CDS and user catalogs under the following conditions:

- You ran EDGHSKP with CATSYNCH and VERIFY, and differences in catalog status were found. DFSMSrmm clears the catalog synchronization date and time to force you to run CATSYNCH.
- DFSMSrmm is not active when catalog activity for tape data sets is taking place. DFSMSrmm issues messages EDG8200E and EDG8201E when DFSMSrmm cannot track catalog updates.
- A DFSMSrmm failure occurs during catalog processing. DFSMSrmm issues messages EDG8200E and EDG8201E when an error is detected in processing.
- You connect or disconnect user catalogs, which contain entries for tape data sets, to the master catalog.

When you use IDCAMS REPRO MERGECAT for catalog maintenance, you do not need to run a job to re-synchronize the DFSMSrmm CDS and catalogs.

Suggestion: Use EDGUTIL with the UPDATE parameter and with the SYSIN command CONTROL CDSID(PROD) CATSYNCH(NO) parameter when you think that the DFSMSrmm CDS and the catalogs are no longer synchronized. This prevents DFSMSrmm from relying on the catalog information in the CDS.

For more details, see the “Maintaining the Control Data Set” chapter in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

9.2 Preparing DFSMSrmm catalog synchronization

The following discussion uses the configuration shown in Figure 9-2. This diagram shows a complex RMMplex implementation with two sysplexes and one non-plex sharing a single DFSMSrmm CDS. The second non-plex shows a normal shared DFSMSrmm CDS. Each plex has its own direct access storage device (DASD) environment that is not shared between the other plexes. Also, each plex has a unique MVS catalog environment.

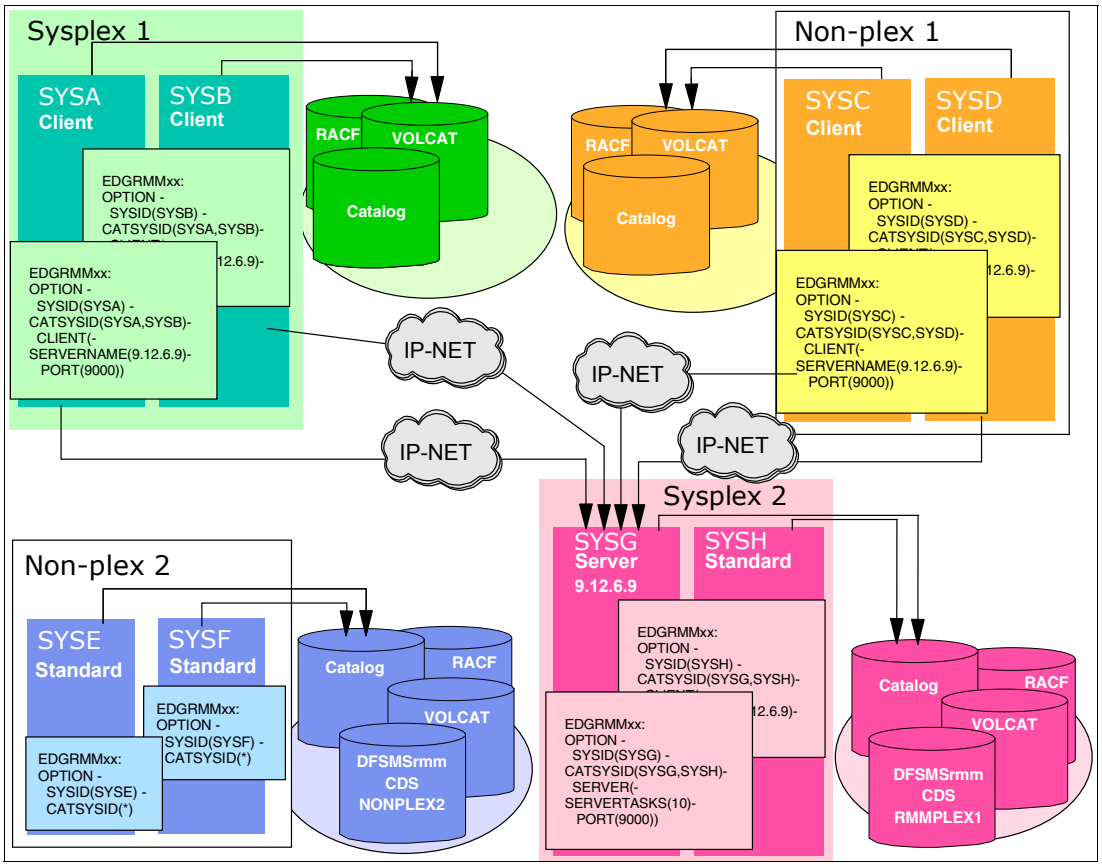


Figure 9-2 RMMplex with six systems sharing one DFSMSrmm CDS

When you enable catalog synchronization, DFSMSrmm synchronizes the DFSMSrmm CDS and available user catalogs. Enabling catalog synchronization is normally a one-time action. When the catalog is synchronized, DFSMSrmm continually tracks and updates the catalog status. You enable catalog synchronization by specifying the DFSMSrmm EDGRMMxx OPTION CATSYSID operand. You can learn more about this operand in the chapter titled “Defining System Options: OPTION” in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Before you start catalog synchronization, check for the following items:

1. Verify that DFSMSrmm has all the authorization to read all integrated catalog facility (ICF) user catalogs including the tape volume catalog (TCDB). Normally, the TCDB is named SYS1.VOLCAT.VGENERAL. If you used a name other than the default, you can find your definition in the DEVSUPnn member in SYS1.SAMPLIB.
2. In an environment with unshared catalogs, make sure that for all current and previously used system IDs, there is an entry in the CATSYSID option of one of the systems.

3. Check for data sets with no system creation ID, because if there is no system creation ID, you can lose data. It is possible to have no system creation ID as a result of the data set records being created during a conversion to RMM using an older version of the conversion tools.

9.2.1 Message data set considerations

Normally, the allocation of the DFSMSrmm message data set is small because you get only a small amount of information if the daily housekeeping is running. If catalog synchronization is running the first time, you see information for each data set that is changed in this run. After conversion, the catalog status of each data set is unknown and is updated to yes or no in this run.

Allocating a larger message data set

You must allocate a larger DFSMSrmm message data set, because the catalog synchronization process writes a record to the DFSMSrmm message file for each data set that is updated in the DFSMSrmm CDS. You can use the JCL shown in Example 9-1 to allocate a larger message data set. If you have catalogs that are not shared, you have to allocate a larger message data set on each single system or plex on which catalog synchronization must run.

Example 9-1 Sample JCL to allocate a larger DFSMSrmm message data set

```
//IEFBR14  EXEC PGM=IEFBR14
//MESSAGE  DD   DISP=(,CATLG,DELETE),
//         DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA),
//         DSN=RMM.HSKP.MESSAGE.CATSYNC,
//         SPACE=(TRK,(100,100))
```

Note: After you check your MESSAGE and ACTIVITY data sets and you successfully synchronize your catalogs, you can delete these files.

9.2.2 Activity log data set considerations (optional)

In a normal daily housekeeping run, you see only a small number of records that tell you the changes made in the vital record processing. During the catalog synchronization process, a record for each data set is changed during the run.

Allocating a larger activity log data set

You must allocate a larger DFSMSrmm activity log data set, because the catalog synchronization process writes a record to the DFSMSrmm message file for each data set that is updated in the DFSMSrmm CDS. You can use the JCL shown in Example 9-2 to allocate a larger message data set. If you have catalogs that are not shared, allocate a larger activity log data set on each single system or plex on which the catalog synchronization must run.

Example 9-2 Sample JCL to allocate a larger DFSMSrmm activity log data set

```
//IEFBR14  EXEC PGM=IEFBR14
//ACTIVITY DD   DISP=(,CATLG,DELETE),
//         DCB=(LRECL=500,BLKSIZE=0,RECFM=VB),
//         DSN=RMM.HSKP.ACTIVITY.CATSYNC,
//         SPACE=(TRK,(100,100))
```

Note: After you check your MESSAGE and ACTIVITY data sets and successfully synchronize your catalogs, you can delete this file.

9.2.3 Journal considerations

The catalog synchronization process also generates many records in the journal data set. Prepare for this by allocating a large journal data set, or process the catalog synchronization run with the journal data set disabled.

Allocating a larger journal data set

Example 9-3 shows sample JCL to create a larger DFSMSrmm journal.

Example 9-3 Sample JCL to create a larger DFSMSrmm journal

```
//STEP02   EXEC PGM=IEFBR14
//SYSPRINT DD   SYSOUT=*
//JOURNAL  DD   DISP=(,CATLG,DELETE),
//         DSN=RMM.PROD.JRNL.LARGE,
//         UNIT=3390,VOL=SER=DFRMM3,
//         AVGREC=U,SPACE=(4096,(15000),,CONTIG)
```

Disabling the journal data set (optional)

You can disable the journal if there are no other tape activities at the time that you are synchronizing your catalogs. The possibility of disabling the journaling depends on how the journal data set is defined in the DFSMSrmm procedure and in the parmlib member EDGRMMxx.

If the journal data set is not defined in the DFSMSrmm procedure, and it is defined in the PARMLIB member EDGRMMxx, it is possible to perform an EDGHSKP PARM(VRSEL) run with the journaling disabled:

1. Create a new member EDGRMMyy by copying the existing member EDGRMMxx in parmlib.
2. In the new member EDGRMMyy, remove the JRNLNAME statement in the OPTION parameter.
3. Check that you have not specified a JOURNAL DD statement in your DFRMM started task job control.
4. Restart DFSMSrmm with the new EDGRMMyy member by issuing the operator MODIFY command as shown in Figure 9-3.

```
F DFRMM,M=yy
```

Figure 9-3 MVS command MODIFY DFRMM

When DFSMSrmm is restarted without the journal data set, you receive the messages shown in Figure 9-4 in your system log.

```
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMMyy IN SYS1.PARMLIB
EDG0122I NO JOURNAL FILE ALLOCATED - JOURNALING DISABLED
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 9-4 DFSMSrmm journal disabled messages

You can also check the status of the journal data set by using the command shown in Figure 9-5.

```
RMM LISTCONTROL CNTL
```

Figure 9-5 RMM LISTCONTROL CNTL subcommand

Example 9-4 shows the output of the LISTCONTROL command.

Example 9-4 LISTCONTROL OPTION output

```
Control record:
Type = MASTER    Create date = 2009/309    Create time = 18:38:11
                 Update date = 2010/282    Update time = 14:30:55
Journal: Utilization =   0% (75% threshold)     STATUS: = DISABLED
CDS:     Utilization =  87%
Exit status:                              Options:
  EDG_EXIT100 = NONE                        Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = ENABLED                     Common Time             = DISABLED
                                            CDSID ENQ name          = ENABLED
Last backup:                              Last expiration processing:
  Date = 2010/282    Time = 14:29:58        Date = 2010/280    Time = 16:33:30
Last journal backup:                      Last store update:
  Date = 2010/282    Time = 14:29:58        Date =             Time =
Last report extract:                      Last VRS processing:
  Date = 2010/280    Time = 16:33:30        Date = 2010/280    Time = 16:33:30
Last scratch procedure:                   Last Catalog synchronize:
  Date =             Time =                 Date =             Time =
Rack numbers       = 6288                 Empty racks        = 6263
LOCAL store bins   = 0                    Empty LOCAL bins   = 0
DISTANT store bins = 0                    Empty DISTANT bins = 0
REMOTE store bins  = 0                    Empty REMOTE bins  = 0
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =
```

The journal file data set name field needs to be blank when the journal data set is disabled.

DFSMSrmm now runs with the journal data set disabled, and the EDGHSKP PARM(VRSEL) job can start. Running EDGHSKP without journaling also reduces the execution time.

9.2.4 Creating system IDs

In an environment with unshared catalogs, check for all current and previously used system IDs. Include them in the CATSYSID values that you must specify in your EDGRMMxx PARMLIB members.

Example 9-5 on page 336 shows sample JCL that can be used to identify all system IDs that were created previously and that are currently in use. A backup copy of the current DFSMSrmm CDS is used as input to gather the necessary information.

Example 9-5 Sample JCL to obtain all system IDs in your RMMplex environment

```
//CRSYSID JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//STEP0001 EXEC PGM=EDGHSKP,
//         PARM=('BACKUP(AMS)')
//SYSPRINT DD   SYSOUT=*
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE,DISP=SHR
//BACKUP   DD   DSN=RMM.HSKP.BACKUP(+1),DISP=(,CATLG),
//         UNIT=SYSDA,SPACE=(CYL,(500,50),RLSE),
//         DCB=(LRECL=9216,BLKSIZE=0,REFM=VB,DSORG=PS)
//JRNLBKUP DD   DSN=RMM.HSKP.JNLBKUP(+1),DISP=(,CATLG),
//         UNIT=SYSDA,SPACE=(CYL,(500,50),RLSE),
//         DCB=(LRECL=9248,BLKSIZE=0,REFM=VB,DSORG=PS)
//* ************************************************************* *
//STEP0002 EXEC PGM=IEBGENER
//SYSPRINT DD   DUMMY
//SYSUT1   DD   DSN=RMM.HSKP.MESSAGE,DISP=SHR
//SYSUT2   DD   SYSOUT=*
//SYSIN    DD   DUMMY
//* ************************************************************* *
//STEP0003 EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=SHR,DSN=RMM.HSKP.BACKUP(+1)
//SORTOUT  DD   DISP=(,CATLG,DELETE),
//         DCB=(LRECL=12,BLKSIZE=0,RECFM=VB),
//         DSN=RMM.HSKP.ALL.CATSYSID,
//         SPACE=(TRK,(1,1))
//SYSIN    DD   *
   SORT FIELDS=(73,8,CH,A),DYNALLOC
   OMIT COND=(5,1,CH,NE,C'D')
   OUTREC FIELDS=(1,4,73,8)
   SUM FIELDS=NONE
/*
```

Note: The DFSMSrmm CDS backup copy must be created by using EDGHSKP,PARM='BACKUP(AMS)'. In a conversion, you can use the file that is used to load the DFSMSrmm control data set.

When the DFSMSrmm journal file is disabled, you must remove the JOURNAL DD statement.

Figure 9-6 shows the result of your previously used SORT. You can see that you have an additional system ID called TEST that was created, and you have data set records without the system information.

```
....
SYSA
SYSB
SYSC
SYSD
SYSG
SYSH
TEST
```

Figure 9-6 List of all used system IDs in the RMMplex environment

Sometimes, you see blank or hex blank lines in the SORTOUT data set. That means that you have data sets without create system ID information.

> **Suggestion:** The create system ID field specifies the SMF system ID of the system where the data set was created. When you run DFSMSrmm with catalogs that are not all fully shared, DFSMSrmm uses the create system ID of the first file on a volume to determine the system where the volume will return to scratch.

For more information about how you can correct this error, see 9.3, “Synchronizing with a shared catalog environment” on page 338 or 9.4, “Synchronizing when catalogs are not shared” on page 345.

9.2.5 Disabling catalog synchronization

Use CATSYNCH(NO) to clear the last synchronization date and time stored in the DFSMSrmm control record. After conversion, the catalog synchronization is normally not switched on. Use the JCL shown in Example 9-6 to disable catalog synchronization.

Example 9-6 Sample JCL to disable catalog synchronization

```
//CATSYNCO JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
   CONTROL CDSID(PROD) CATSYNCH(NO)
/*
```

You can verify that the catalog synchronization is disabled by using the TSO RMM LISTCONTROL subcommand shown in Figure 9-7.

```
TSO RMM LISTCONTROL
```

Figure 9-7 RMM LISTCONTROL subcommand to check catsync status

Example 9-7 shows the result of the TSO RMM LISTCONTROL subcommand. The last catalog synchronization date and time fields are blank because the function is disabled.

Example 9-7 Output of the TSO RMM LISTCONTROL subcommand

```
Control record:
Type = MASTER    Create date = 2009/309    Create time = 18:38:11
                 Update date = 2010/282    Update time = 14:30:55
Journal: Utilization =   0% (75% threshold)     STATUS: = DISABLED
CDS:     Utilization =  87%
Exit status:                              Options:
  EDG_EXIT100 = NONE                        Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = ENABLED                     Common Time             = DISABLED
                                            CDSID ENQ name          = ENABLED
Last backup:                              Last expiration processing:
  Date = 2010/282    Time = 14:29:58        Date = 2010/280    Time = 16:33:30
Last journal backup:                      Last store update:
  Date = 2010/282    Time = 14:29:58        Date =             Time =
Last report extract:                      Last VRS processing:
```

```
  Date = 2010/280    Time = 16:33:30      Date = 2010/280    Time = 16:33:30
Last scratch procedure:                 Last Catalog synchronize:
  Date =             Time =               Date =             Time =
Rack numbers       = 6288               Empty racks        = 6263
LOCAL store bins   = 0                  Empty LOCAL bins   = 0
DISTANT store bins = 0                  Empty DISTANT bins = 0
REMOTE store bins  = 0                  Empty REMOTE bins  = 0
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =
```

If you have an environment with a shared catalog environment, you can continue to 9.3, “Synchronizing with a shared catalog environment” on page 338. If you have an environment where not all the catalogs are shared, continue with 9.4, “Synchronizing when catalogs are not shared” on page 345.

9.3 Synchronizing with a shared catalog environment

This section explains how you can implement catalog synchronization in an environment with all catalogs shared as shown in Figure 9-8. You can run the jobs on any of the systems sharing the catalog, but you must update the EDGRMMxx parmlib member on each system sharing the catalog. In our example, EDGRMMxx is updated on both SYSE and SYSF.

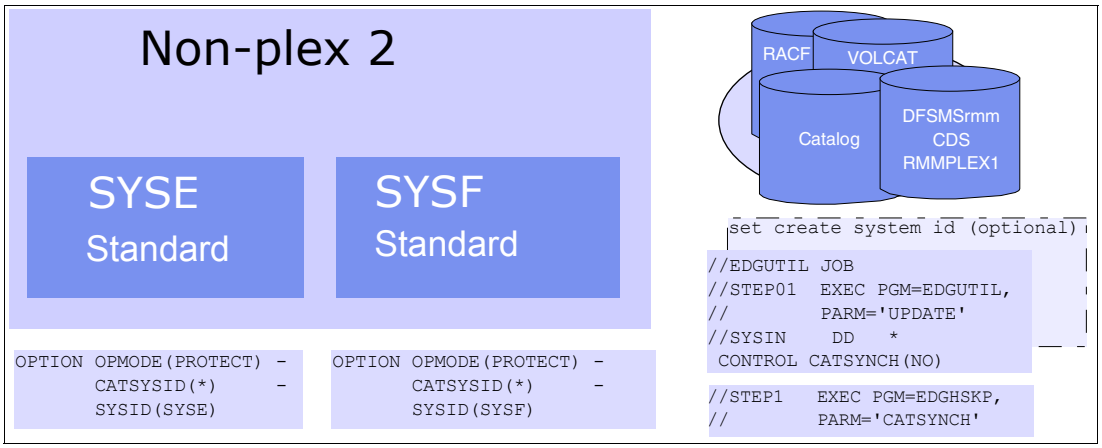


Figure 9-8 Flow to synchronize with a shared catalog environment

You can use shared user catalogs regardless of whether the CDS and user catalogs are synchronized. Synchronize the DFSMSrmm CDS and user catalogs so that DFSMSrmm does not need to locate catalog information for every data set that is managed using catalog control.

Follow these steps to synchronize your catalogs in a fully shared catalog environment:

1. Correct missing create system ID information (optional).
2. Allocate a larger message data set. See 9.2.1, “Message data set considerations” on page 333 for more information.

3. Allocate a larger activity log data set as described in “Activity log data set considerations (optional)” on page 333.
4. Run the EDGUTIL UPDATE with the SYSIN command to clear catalog synchronization using the job control shown in Figure 9-9.

```
//CATSYNCO JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
   CONTROL CDSID(PROD) CATSYNCH(NO)
/*
```

Figure 9-9 EDGUTIL to clear catalog synchronization

5. Specify the DFSMSrmm EDGRMMxx parmlib OPTION on all systems as shown in Figure 9-10.

```
OPTION    OPMODE(R)                    /* Record-Only Mode         */-
          ACCOUNTING(J)                /* Accounting from JOB      */-
          BACKUPPROC(EDGBKUP)          /* Name of BACKUP-proc      */-
          BLP(RMM)                     /* DFSMSrmm controls BLP    */-
          CATRETPD(0012)               /* catalog retention        */-
          CATSYSID(*)                  /* all catalogs shared      */-
          CDSID(PROD)                  /* control data set id      */-
...
```

Figure 9-10 EDGRMMnn parmlib member including the CATSYSID command

6. Restart DFSMSrmm using the MVS MODIFY command as shown in Figure 9-11.

```
MODIFY DFRMM,M=01
   or
F DFRMM,M=01
```

Figure 9-11 Restart DFRMM

7. Run the DFSMSrmm EDGHSKP utility using the JCL shown in Figure 9-12.

```
//LCLHSKP  JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGHSKP,PARM=’CATSYNCH’
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.CATSYNC,DISP=SHR
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY.CATSYNC,DISP=SHR
```

Figure 9-12 EDGRMMnn parmlib member including the CATSYSID command

9.3.1 Data sets without create system ID information

If you have data sets without the create system ID information, and all user catalogs are shared, you can use the JCL shown in Example 9-8 to list all data sets without create system ID information and to correct these errors.

In this case, you can use the same create system ID for all data sets. We used the OUTREC function to create TSO RMM CHANGEDATASET subcommands directly. In the third step, we run the created commands to correct the errors.

Example 9-8 Sample JCL to list all data sets without system creation ID information

```
//CRSYSID JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD   DUMMY
//SYSIN    DD   *
  DEL RMM.HSKP.WITHOUT.CATSYSID NONVSAM PURGE
  SET MAXCC=0
/*
//* ******************************************************* *
//STEP02   EXEC PGM=EDGHSKP,PARM=(',RPTEXT,DATEFORM(J)')
//SYSPRINT DD   SYSOUT=*
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.DSET,DISP=SHR
//XREPTEXT DD   DSN=RMM.HSKP.EXTRACTX,DISP=SHR
//* ******************************************************* *
//SORT1    EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=SHR,DSN=RMM.HSKP.EXTRACTX
//SORTOUT  DD   DISP=(,PASS,DELETE),
//         DSN=&&SORT1,
//         DCB=*.SORTIN,
//         SPACE=(CYL,(50,50))
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
     INCLUDE COND=(809,44,CH,NE,C' ',AND,5,1,CH,EQ,C'X')
     OPTION VLSHRT
/*
//* ******************************************************* *
//SORT2    EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=(OLD,DELETE),DSN=&&SORT1
//SORTOUT  DD   DISP=(,CATLG,DELETE),
//         DCB=(LRECL=105,BLKSIZE=0,RECFM=VB),
//         DSN=RMM.HSKP.WITHOUT.CATSYSID,
//         SPACE=(TRK,(1,1))
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
     INCLUDE COND=((809,44,CH,NE,C' '),AND,
                   869,8,CH,EQ,C'        ',OR,
                   869,8,CH,EQ,X'0000000000000000')
     OUTREC FIELDS=(1,4,C'          RMM CD ',809,44,
                    C' VOL(',9,6,C') SYSID(SYSE) SEQ(',
                    915,4,C') FORCE')
     OPTION VLSHRT
/*
//* ******************************************************* *
//CORREC   EXEC PGM=IKJEFT01
//SYSOUT   DD   SYSOUT=*
```

```
//SYSTSPRT DD   SYSOUT=*
//SYTSIN   DD   DSN=RMM.HSKP.WITHOUT.CATSYSID,DISP=SHR
```

Note: You only need to verify that the first data set on each volume has a create system ID, because that is the only data set that RMM EXPROC uses. This can result in a much shorter list of data sets to check and commands to run.

9.3.2 Clearing the catalog synchronization

Run the EDGUTIL UPDATE with the SYSIN command CONTROL CDSID(PROD) CATSYNCH(NO) as shown in Example 9-9 to clear the last synchronization date and time.

Note: You can run the EDGUTIL at any time to clear the synchronization date and time whether this information was previously set or not.

Example 9-9 Sample JCL to clear catalog synchronization

```
//LCLUTILC JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 CONTROL CDSID(PROD) CATSYNCH(NO)
/*
```

9.3.3 Updating the EDGRMMxx DFSMSrmm options on all systems

Update the EDGRMMxx parmlib member for DFSMSrmm (Example 9-10 on page 342 and Example 9-11 on page 342) to include the CATSYSID operand.

CATSYSID

Specify CATSYSID to enable DFSMSrmm catalog synchronization. Use the CATSYSID operand to identify the user catalogs you want tracked when you run the DFSMSrmm EDGHSKP utility with the CATSYNCH operand to exploit catalog status tracking. All the systems that you identify must have the code that supports catalog synchronization.

CATSYSID(*) means that all catalogs are fully shared. You must specify an asterisk (*) to specify that catalogs are fully shared so that any data set can be processed by DFSMSrmm on any DFSMSrmm subsystem.

CATSYSID(sys_ID_list) provides a list of DFSMSrmm system IDs that share tape data set user catalogs with this DFSMSrmm subsystem. You can identify up to 16 system IDs for DFSMSrmm subsystems. You must specify a list of system IDs when user catalogs are not shared. You must also ensure that they are synchronized with the DFSMSrmm CDS before you run inventory management vital record processing. Include IDs for systems that you no longer use but for which you still retain tape data sets.

The default is None.

Update the EDGRMMxx parmlib member on system SYSE.

Example 9-10 EDGRMMxx parmlib member for system SYSE

```
OPTION OPMODE(P)                        /* operating mode      */ -
       ....                                                       -
       CATSYSID(*)                      /* shared catalogs     */ -
       SYSID(SYSE)                      /* system ID           */ -
       ....
```

Update the EDGRMM PARMLIB member on system SYSF.

Example 9-11 EDGRMMxx parmlib member for system SYSF

```
OPTION OPMODE(P)                        /* operating mode      */ -
       ....                                                       -
       CATSYSID(*)                      /* shared catalogs     */ -
       SYSID(SYSF)                      /* system ID           */ -
       ....
```

When the CATSYSID operand is specified in the EDGRMMxx parmlib member, restart DFRMM on system SYSE and SYSF to activate this new definition using the command shown in Figure 9-13.

```
MODIFY DFRMM,M=00
   or
F DFRMM,M=00
```

Figure 9-13 Modify DFSMSrmm

After you enter this command, the message shown in Figure 9-14 displays.

```
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 9-14 EDG0105I initialization message

9.3.4 Synchronizing your catalogs

Run the DFSMSrmm EDGHSKP utility with the PARM=CATSYNCH parameter as shown in Example 9-12. Do this once on each plex to synchronize all data set records in the CDS with the status from the catalog. Before you run CATSYNCH, allocate larger MESSAGE (see Example 9-1 on page 333) and ACTIVITY (see Example 9-2 on page 333) data sets that are large enough to contain all the messages that are issued during the catalog synchronization process.

Note: Also, the journal data set needs to be large or disabled at this time.

For example, consider when you run CATSYNCH for the first time, perhaps as part of DFSMSrmm implementation, or you run CATSYNCH with VERIFY. In this case, expect messages for each data set that must be updated. Otherwise, the status is different between the catalogs and the DFSMSrmm CDS.

Example 9-12 Sample JCL to synchronize your shared catalogs

```
//LCLHSKP  JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGHSKP,PARM='CATSYNCH'
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.CATSYNC,DISP=SHR
```

```
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY.CATSYNC,DISP=SHR
```

Note: You can run the job on any of the systems, SYSE or SYSF.

9.3.5 Checking catalog synchronization

Display DFSMSrmm control information as shown in Figure 9-15 to check whether the DFSMSrmm control data set and system catalogs are synchronized.

```
TSO RMM LISTCONTROL
```

Figure 9-15 Check catalog synchronization setting

Example 9-13 shows the output from the RMM LISTCONTROL command.

Example 9-13 Sample LISTCONTROL subcommand output

```
Control record:
Type = MASTER    Create date = 2009/309    Create time = 18:38:11
                 Update date = 2010/282    Update time = 14:30:55
Journal: Utilization =   0% (75% threshold)     STATUS: = DISABLED
CDS:     Utilization =  87%
Exit status:                              Options:
  EDG_EXIT100 = NONE                        Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = ENABLED                     Common Time             = DISABLED
                                            CDSID ENQ name          = ENABLED
Last backup:                              Last expiration processing:
  Date = 2010/282    Time = 14:29:58        Date = 2010/280    Time = 16:33:30
Last journal backup:                      Last store update:
  Date = 2010/282    Time = 14:29:58        Date =             Time =
Last report extract:                      Last VRS processing:
  Date = 2010/280    Time = 16:33:30        Date = 2010/280    Time = 16:33:30
Last scratch procedure:                   Last Catalog synchronize:
  Date =             Time =                 Date = 2010/280    Time = 16:58:37
Rack numbers       = 6288                 Empty racks        = 6263
LOCAL store bins   = 0                    Empty LOCAL bins   = 0
DISTANT store bins = 0                    Empty DISTANT bins = 0
REMOTE store bins  = 0                    Empty REMOTE bins  = 0
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =
```

Note: When running EDGHSKP expiration processing or vital record processing, CATSYSID(*) might have been specified in the EDGRMMxx parmlib member and the CDS might not be synchronized with the catalogs. In this case, DFSMSrmm initiates catalog synchronization even if you have not specified it.

Look at the message data set. You can see all data sets that are updated in the DFSMSrmm CDS by this run. Example 9-14 shows an example.

Example 9-14 Message data set after catalog synchronization

```
EDG6001I INVENTORY MANAGEMENT STARTING ON 2010/284 AT 14:04:47 -
         PARAMETERS IN USE ARE
         DATEFORM(J),CATSYNCH,DATE(2010/284)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
         VRSEL(NEW)
         VRSJOBNAME(2)
         VRSMIN(1,INFO)
         VRSCHANGE(INFO)
         VRSDROP(PERCENT(10),INFO) VRSRETAIN(PERCENT(80),INFO)
                 EXPDTDROP(PERCENT(10),INFO)
         SMSTAPE(PURGE(ASIS) UPDATE(EXITS,SCRATCH,COMMAND))
         CATRETPD(6)
         UNCATALOG(Y)
         TPRACF(N)
         NOTIFY(N)
         SYSID(SC70)
         CATSYSID(*)
         RETAINBY(SET)
         MOVEBY(SET)
         GDG(CYCLEBY(GENERATION),DUPLICATE(BUMP))
EDG2234I DFSMSrmm CDS CATALOG STATUS UNKNOWN FOR SCHLUM.RMMDEMO.FILE4.VOL23
         VOLUM.RMMDEMO.FILE4.VOL23 VOLUME SC0010 FILE 3 SYNCHRONIZED TO
         CATALOG STATUS
EDG2234I CONT:-  YES
....
EDG2234I DFSMSrmm CDS CATALOG STATUS YES FOR WWZ038.GSD1997A.DUMP VOLUME
         SC0005 FILE 1 SYNCHRONIZED TO CATALOG STATUS NO
EDG2234I DFSMSrmm CDS CATALOG STATUS YES FOR WWZ038.GSD1997A.DUMP VOLUME
         SC0011 FILE 1 SYNCHRONIZED TO CATALOG STATUS NO
....
EDG2234I DFSMSrmm CDS CATALOG STATUS NO FOR WWZ038.SAVE.ZOS120.DU01A426
         VOLUME Q17062 FILE 1 SYNCHRONIZED TO CATALOG STATUS YES
EDG2234I DFSMSrmm CDS CATALOG STATUS NO FOR WWZ038.SAVE.ZOS120.DU01A483
         VOLUME Q17030 FILE 1 SYNCHRONIZED TO CATALOG STATUS YES
EDG2307I INVENTORY MANAGEMENT TASK CATSYNCH COMPLETED SUCCESSFULLY
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

9.3.6 Enabling the journal data set (optional)

When the EDGHSKP run is complete, activate the journal data set again. To activate the journal data set, follow these steps:

1. Restart DFSMSrmm with the EDGRMMxx PARMLIB member, by using the command shown in Figure 9-16.

```
F DFRMM,M=xx
```

Figure 9-16 Restart DFSMSrmm subsystem

2. Back up the DFSMSrmm CDS and the journal data set, with the EDGHSKP PARM='BACKUP' utility. The DFSMSrmm journal data set is now active, nullified, and synchronized with the DFSMSrmm CDS after the backup. You see the messages shown in Figure 9-17 on page 345.

```
EDG2121I DFSMSrmm JOURNAL FILE IS NOW CLEARED
EDG2121I DFSMSrmm JOURNAL FILE IS NOW ENABLED
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

Figure 9-17 DFSMSrmm journal enable messages

Note: If the journal was previously used, you must delete it and allocate a new one, because the CDS and JOURNAL control information no longer match.

9.3.7 Catalog re-synchronization with a shared catalog environment

To request catalog re-synchronization, repeat the steps between 9.2.5, “Disabling catalog synchronization” on page 337 and 9.3.6, “Enabling the journal data set (optional)” on page 344.

Note: Rerun catalog synchronization if you have manually added volumes in MASTER or USER status with data set information, or you are changing your catalog environment (MERGECAT, IMPORT, or EXPORT).

9.4 Synchronizing when catalogs are not shared

This section explains how you can implement catalog synchronization in an environment when catalogs are not shared as shown in Figure 9-18 on page 346. You must run the synchronization jobs on each PLEX once, but the update of the DFSMSrmm control record can run on any system. You must also update the EDGRMMxx parmlib member on each system sharing the catalog. In our example, EDGRMMxx is updated:

- On both SYSA and SYSB with CATSYSID(SYSA,SYSB)
- On both SYSG and SYSH with CATSYSID(SYSG,SYSH)
- On both SYSC and SYSD with CATSYSID(SYSC,SYSD)

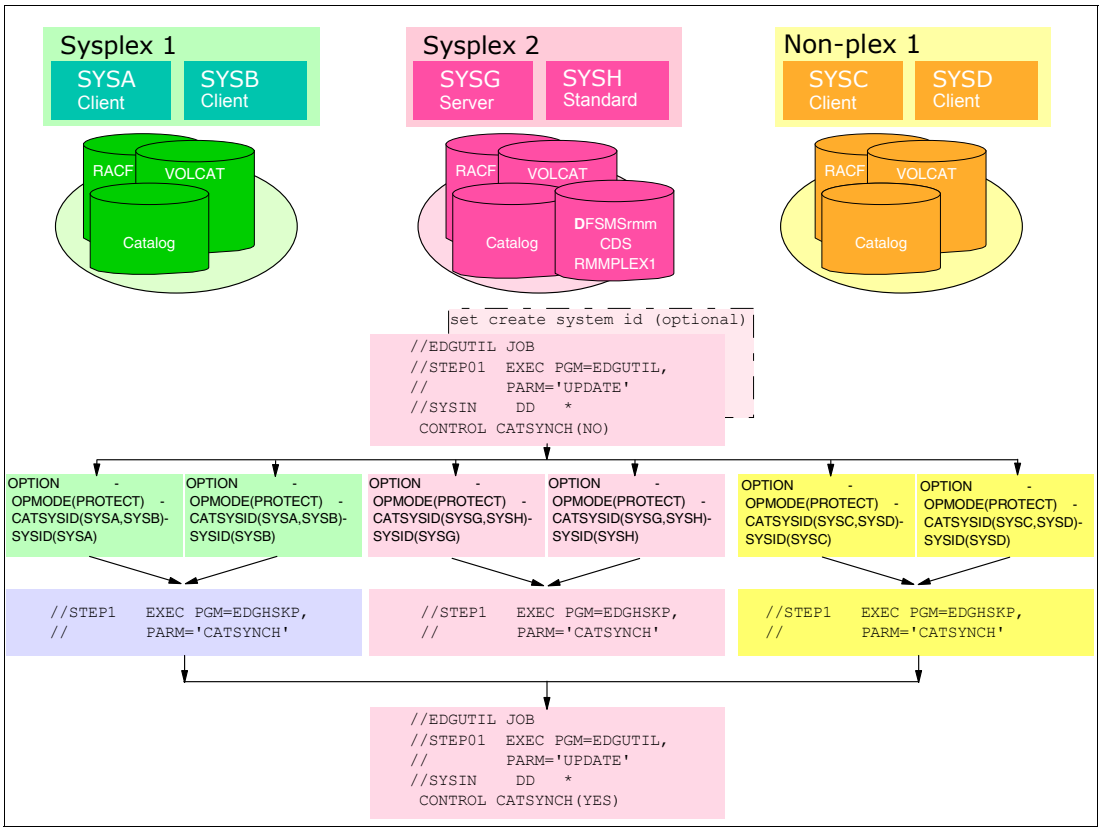


Figure 9-18 Job flow to synchronize when catalogs are not shared

When catalogs are not fully shared, you can have non-generation data group (GDG) and GDG data sets, with the same name, that are cataloged but in different, unshared catalogs. To retain all copies of all data sets, define a retention policy that uses DAYS with COUNT(99999) rather than CYCLES. Using CYCLES can produce unpredictable results, for example, the premature expiration of data sets.

To ensure cycles are grouped together, use the job name to further qualify the data sets, using a different job name for each set of cataloged data sets. For example, use JOBNAME(xjzA) for data sets created on system SYSA, and JOBNAME(xjzH) for data sets created on system SYSH. This enables the cycle retention to be based on the scope of the creation system.

Follow these tasks, which are also shown in Figure 9-18, to synchronize your catalogs when your catalog environment is not shared:

1. Correct missing create system ID information (optional).
2. Allocate a larger message data set and a larger activity log data set.
3. Run the EDGUTIL UPDATE with the SYSIN command to clear catalog synchronization using the job control shown in Figure 9-19 on page 347.

```
//CATSYNCO JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
   CONTROL CDSID(PROD) CATSYNCH(NO)
/*
```

Figure 9-19 EDGUTIL to clear catalog synchronization

4. Specify the DFSMSrmm EDGRMMxx PARMLIB OPTION on all systems as shown in Figure 9-20.

```
OPTION    OPMODE(R)                    /* Record-Only Mode         */-
          ACCOUNTING(J)                /* Accounting from JOB      */-
          BACKUPPROC(EDGBKUP)          /* Name of BACKUP-proc      */-
          BLP(RMM)                     /* DFSMSrmm controls BLP    */-
          CATRETPD(0012)               /* catalog retention        */-
          CATSYSID(SYSA,SYSB)          /* catalogs not shared      */-
          CDSID(PROD)                  /* control data set id      */-
...
```

Figure 9-20 EDGRMMxx parmlib member including CATSYSID command

5. Restart DFSMSrmm using the MVS MODIFY command in Figure 9-21.

```
MODIFY DFRMM,M=01
   or
F DFRMM,M=01
```

Figure 9-21 Restart DFRMM

6. Run the DFSMSrmm EDGHSKP utility using the JCL shown in Figure 9-22.

```
//STEP01   EXEC PGM=EDGHSKP,PARM=’CATSYNCH’
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.CATSYNC,DISP=SHR
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY.CATSYNC,DISP=SHR
```

Figure 9-22 EDGRMMxx parmlib member includes the CATSYSID command

7. Run the EDGUTIL UPDATE using the SYSIN command as shown in Figure 9-23.

```
//CATSYNCO JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//UTIL     EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN    DD   *
   CONTROL CDSID(PROD) CATSYNCH(YES)
/*
```

Figure 9-23 EDGUTIL to set catalog synchronization

9.4.1 Setting the system creation ID if catalogs are not shared

You can use the information in this section if you have either a normal DFSMSrmm environment where catalogs are not fully shared or if you have implemented an RMMplex with catalogs that are not fully shared. Example 9-15 shows sample JCL that can be used to set a system ID for all data sets on volumes 5* and 9* without having to create a set of system IDs on which the volumes were created if you have catalogs that are not fully shared. It also shows how to determine all data sets without create system ID information residing in different volume pools.

TSO RMM CHANGEDATASET subcommands are created with different system creation IDs (these are the IDs of the systems on which the volumes were created) to correct the errors. All volumes with a volume prefix of 5 and 9 are used on SYSA, and all other volumes are used on SYSG.

Example 9-15 Sample JCL to determine all data sets without system creation ID details

```
//CRSYSID JOB (DEOA047),SCHLUMBERGER,MSGLEVEL=1,MSGCLASS=H,
//             REGION=4M,NOTIFY=&SYSUID
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD   DUMMY
//SYSIN    DD   *
  DEL RMM.HSKP.WITHOUT.CATSYSID.SYSA NONVSAM PURGE
  DEL RMM.HSKP.WITHOUT.CATSYSID.SYSG NONVSAM PURGE
  SET MAXCC=0
/*
//* ******************************************************* *
//STEP02   EXEC PGM=EDGHSKP,PARM=(',RPTEXT,DATEFORM(J)')
//SYSPRINT DD   SYSOUT=*
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.DSET,DISP=SHR
//XREPTEXT DD   DSN=RMM.HSKP.EXTRACTX,DISP=SHR
//* ******************************************************* *
//SORT1    EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=SHR,DSN=RMM.HSKP.EXTRACTX
//SORTOUT  DD   DISP=(,PASS,DELETE),
//         DSN=&&SORT1,
//         DCB=*.SORTIN,
//         SPACE=(CYL,(50,50))
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
     INCLUDE COND=(809,44,CH,NE,C' ')
     OPTION VLSHRT
/*
//* ******************************************************* *
//SORT2    EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=(OLD,PASS),DSN=&&SORT1
//SORTOUT  DD   DISP=(,CATLG,DELETE),
//         DCB=*.SORTIN,
//         DSN=RMM.HSKP.WITHOUT.CATSYSID.SYSA,
//         SPACE=(TRK,(10,1))
//*        ************************************************
//*        * include all volumes starting with 5 and 9     *
//*        ************************************************
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
     INCLUDE COND=((9,1,CH,EQ,C'5',OR,9,1,CH,EQ,C'9'),AND,
                  (869,8,CH,EQ,C'        ',OR,
                   869,8,CH,EQ,X'0000000000000000'))
```

```
     OPTION VLSHRT
     OUTREC FIELDS=(1,4,C'          RMM CD ',809,44,
                    C' VOL(',9,6,C') SYSID(SYSA) SEQ(',
                    915,4,C') FORCE')
/*
//* ***************************************************** *
//SORT3    EXEC PGM=SORT
//SYSOUT   DD   SYSOUT=*
//SORTIN   DD   DISP=(OLD,DELETE),DSN=&&SORT1
//SORTOUT  DD   DISP=(,CATLG,DELETE),
//         DCB=*.SORTIN,
//         DSN=RMM.HSKP.WITHOUT.CATSYSID.SYSG,
//         SPACE=(TRK,(1,1))
//*        ************************************************
//*        * exclude all volumes starting with 5 and 9     *
//*        ************************************************
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
     INCLUDE COND=((9,1,CH,NE,C'5',AND,9,1,CH,NE,C'9'),AND,
                  (869,8,CH,EQ,C'        ',OR,
                   869,8,CH,EQ,X'0000000000000000'))
     OPTION VLSHRT
     OUTREC FIELDS=(1,4,C'          RMM CD ',809,44,
                    C' VOL(',9,6,C') SYSID(SYSG) SEQ(',
                    915,4,C') FORCE')
/*
//* ***************************************************** *
//CORREC   EXEC PGM=IKJEFT01
//SYSOUT   DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYTSIN   DD   DSN=RMM.HSKP.WITHOUT.CATSYSID.SYSA,DISP=SHR
//         DD   DSN=RMM.HSKP.WITHOUT.CATSYSID.SYSG,DISP=SHR
```

Figure 9-24 shows the created TSO RMM CHANGEDATASET subcommands to correct all data set records that do not have the creating-system information available.

```
RMM CD TAPEA.SAVE.DUMP.D000630 VOL(901254) SYSID(SYSA) SEQ(   1) FORCE
....
RMM CD SYSG.SYSOUT.SAVE.D2004 VOL(A05372) SYSID(SYSG) SEQ(   1) FORCE
```

Figure 9-24 TSO RMM subcommands to update data set records

Each system ID used in your environment, as shown in Figure 9-8 on page 338, must be defined in one of your EDGRMMnn PARMLIB members. A volume with data sets on it with a create system ID not defined in any of the CATSYSID operands in the EDGRMMnn PARMLIB members never returns to status *scratch*. The expiration processing checks that the create system of the volume matches one of the system IDs defined in the CATSYSID PARMLIB option.

You can use the same procedure to set a system ID for all data sets that have no system ID set to change the create system ID from an old system ID to a current one. For example, you can change all data sets that have a create system ID of TEST to a create system ID of SYSH. To do this, change the statement as shown in Example 9-16.

Example 9-16 Sort statements to change the create system ID

```
//SYSIN    DD   *
     SORT FIELDS=(809,44,CH,A,915,4,CH,A),DYNALLOC
```

```
      INCLUDE COND=((9,1,CH,EQ,C'5',OR,9,1,CH,EQ,C'9'),AND,
                   (869,8,CH,EQ,C'TEST    '))
      OPTION VLSHRT
      OUTREC FIELDS=(1,4,C'          RMM CD ',809,44,
                     C' VOL(',9,6,C') SYSID(SYSH) SEQ(',
                     915,4,C') FORCE')
/*
```

Important: If you have data set records without the creating system information or with create system information not defined in any of the parmlib members, these volumes never return to scratch.

9.4.2 Clearing the catalog synchronization

Run the EDGUTIL UPDATE with the SYSIN CONTROL CATSYNCH(NO) command as shown in Example 9-17 to clear the last synchronization date and time. You can run this job on any system in the sysplex. In our case, we ran this job on one of the systems in SYSPLEX2 because these systems have direct access to the DFSMSrmm control data set.

Note: You can run the EDGUTIL at any time to clear the synchronization date and time, whether or not this information was previously set.

Example 9-17 Sample JCL to clear catalog synchronization

```
//LCLUTILC JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 CONTROL CDSID(PROD) CATSYNCH(NO)
/*
```

9.4.3 Updating the EDGRMMxx DFSMSrmm options

To ensure that DFSMSrmm processes the return to scratch, you must specify the EDGRMMxx PARMLIB OPTION CATSYSID operand with the system IDs on which the volumes were created. DFSMSrmm uses the CATSYSID operand to determine the systems to which volumes can be returned to scratch status.

DFSMSrmm checks the system creation ID for the first file on the volume with the list of IDs that are specified for the CATSYSID operand. If there is a match, DFSMSrmm processes the return to scratch. Update the EDGRMMxx PARMLIB member as shown in Example 9-18 on page 351 to Example 9-22 on page 352 to include the CATSYSID operand on each system sharing this DFSMSrmm control data set.

CATSYSID

Specify CATSYSID to enable DFSMSrmm catalog synchronization. Use the CATSYSID operand to identify the user catalogs you want tracked when you run the DFSMSrmm EDGHSKP utility with the CATSYNCH operand to exploit catalog status tracking. All the systems that you identify must have the code that supports catalog synchronization.

CATSYSID(*) All catalogs are fully shared. You must specify an asterisk (*) to specify that catalogs are fully shared so that any data set can be processed by DFSMSrmm on any DFSMSrmm subsystem.

CATSYSID(*sys_ID_list*)

Provides a list of DFSMSrmm system IDs that share tape data set user catalogs with this DFSMSrmm subsystem. You can identify up to 16 system IDs for DFSMSrmm subsystems. Specify a list of system IDs when user catalogs are not shared. Ensure that they are synchronized with the DFSMSrmm CDS before you run inventory management vital record processing. Include IDs for systems that you no longer use but for which you still retain tape data sets.

The default is None.

Example 9-18 through Example 9-22 on page 352 show the correct update of the CATSYSID and optionally the SYSID of each of the six systems that share this DFSMSrmm control data set.

Update the EDGRMM parmlib member on system SYSA.

Example 9-18 EDGRMMxx parmlib member for system SYSA

```
OPTION OPMODE(P)                        /* operating mode         */ -
       ....                                                         -
       CDSID(RMMPLEXP)                  /* DFSMSrmm CDS ID (PROD) */ -
       CATSYSID(SYSA,SYSB)              /* unshared catalogs      */ -
       SYSID(SYSA)                      /* system ID              */ -
       ....                                                         -
```

Update the EDGRMM parmlib member on system SYSB.

Example 9-19 EDGRMMxx parmlib member for system SYSB

```
OPTION OPMODE(P)                        /* operating mode         */ -
       ....                                                         -
       CDSID(RMMPLEXP)                  /* DFSMSrmm CDS ID (PROD) */ -
       CATSYSID(SYSA,SYSB)              /* unshared catalogs      */ -
       SYSID(SYSA)                      /* system ID              */ -
       ....                                                         -
```

Update the EDGRMM parmlib member on system SYSC.

Example 9-20 EDGRMMxx parmlib member for system SYSC

```
OPTION OPMODE(P)                        /* operating mode         */ -
       ....                                                         -
       CDSID(RMMPLEXP)                  /* DFSMSrmm CDS ID (PROD) */ -
       CATSYSID(SYSC,SYSD)              /* unshared catalogs      */ -
       SYSID(SYSC)                      /* system ID              */ -
       ....                                                         -
```

Update the EDGRMM parmlib member on system SYSD.

Example 9-21 EDGRMMxx parmlib member for system SYSD

```
OPTION OPMODE(P)                        /* operating mode         */ -
       ....                                                         -
       CDSID(RMMPLEXP)                  /* DFSMSrmm CDS ID (PROD) */ -
       CATSYSID(SYSC,SYSD)              /* unshared catalogs      */ -
       SYSID(SYSD)                      /* system ID              */ -
       ....                                                         -
```

Update the EDGRMM parmlib member on system SYSG.

Example 9-22 EDGRMMxx parmlib member for system SYSG

```
OPTION OPMODE(P)                        /* operating mode         */ -
       ....                                                          -
       CDSID(RMMPLEXP)                  /* DFSMSrmm CDS ID (PROD) */ -
       CATSYSID(SYSG,SYSH)              /* unshared catalogs      */ -
       SYSID(SYSG)                      /* system ID              */ -
       ....                                                          -
```

Update the EDGRMM parmlib member on system SYSH.

When the CATSYSID operand is specified in the EDGRMMxx parmlib member, restart DFRMM on system SYSA and SYSB, SYSC and SYSD, and SYSG and SYSH as shown in Figure 9-25 to activate this new definition.

```
MODIFY DFRMM,M=00
    or
F DFRMM,M=00
```

Figure 9-25 Restart the DFSMSrmm subsystem

After you enter this command, you see the message shown in Figure 9-26 on each system.

```
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 9-26 DFSMSrmm subsystem completion message

9.4.4 Synchronizing your catalogs

Run the DFSMSrmm EDGHSKP utility with the PARM=CATSYNCH parameter as shown in Example 9-23 on each plex (SYSPLEX1, SYSPLEX2, and non-PLEX1). Do this one time to synchronize all data set records in the CDS with the status from the catalog. Before you run CATSYNCH, allocate larger MESSAGE (see Example 9-1 on page 333) and ACTIVITY (see Example 9-2 on page 333) data sets that are large enough to contain all the messages that are issued during the catalog synchronization process. Also, check the size of your journal or disable it.

The DFSMSrmm subsystem writes a message to each of these data sets for each record that is updated in this run. For example, you can run CATSYNCH for the first time, perhaps as part of DFSMSrmm implementation, or you can run CATSYNCH with VERIFY. In this case, expect messages for each data set that must be updated. Otherwise, the status differs between the catalogs and the DFSMSrmm CDS.

Example 9-23 Sample JCL to synchronize your catalogs that are not shared

```
//LCLHSKP  JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGHSKP,PARM='CATSYNCH'
//MESSAGE  DD   DSN=RMM.HSKP.MESSAGE.CATSYNC,DISP=SHR
//ACTIVITY DD   DSN=RMM.HSKP.ACTIVITY.CATSYNC,DISP=SHR
```

You have to run the job once on sysplex 1 (systems SYSA or SYSB), once on non-plex 1 (systems SYSC or SYSD), and finally once on sysplex 2 (systems SYSG or SYSH) to synchronize all catalogs currently used in this RMMplex.

Ensure that all jobs run without any errors. Also, check your MESSAGE files and ACTIVITY logs for error or warning messages. After you perform these steps, run EDGUTIL with PARM=UPDATE and the sysin operand CONTROL CATSYNCH(YES). This sets the last catalog synchronization date and time in the DFSMSrmm CDS control record as shown in Example 9-24.

Example 9-24 Sample JCL to set the last catalog synchronization date and time

```
//LCLUTILC JOB ,140.SCHLUMBERGER,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1)
//STEP01   EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 CONTROL CDSID(PROD) CATSYNCH(YES)
/*
```

Display DFSMSrmm control information as shown in Figure 9-27 to check whether the DFSMSrmm control data set and system catalogs are synchronized.

```
TSO RMM LISTCONTROL
```

Figure 9-27 Check catalog synchronization setting

Example 9-25 shows the output from the RMM LISTCONTROL command.

Example 9-25 Sample LISTCONTROL subcommand output

```
Control record:
Type = MASTER    Create date = 2009/309    Create time = 18:38:11
                 Update date = 2010/282    Update time = 14:30:55
Journal: Utilization =   0% (75% threshold)     STATUS: = DISABLED
CDS:     Utilization =  87%
Exit status:                              Options:
  EDG_EXIT100 = NONE                        Stacked Volumes         = NONE
  EDG_EXIT200 = NONE                        Extended Bin            = DISABLED
  EDG_EXIT300 = ENABLED                     Common Time             = DISABLED
                                            CDSID ENQ name          = ENABLED
Last backup:                              Last expiration processing:
  Date = 2010/282    Time = 14:29:58        Date = 2010/280    Time = 16:33:30
Last journal backup:                      Last store update:
  Date = 2010/282    Time = 14:29:58        Date =             Time =
Last report extract:                      Last VRS processing:
  Date = 2010/280    Time = 16:33:30        Date = 2010/280    Time = 16:33:30
Last scratch procedure:                   Last Catalog synchronize:
  Date =             Time =                 Date = 2010/280    Time = 16:58:37
Rack numbers       = 6288                 Empty racks        = 6263
LOCAL store bins   = 0                    Empty LOCAL bins   = 0
DISTANT store bins = 0                    Empty DISTANT bins = 0
REMOTE store bins  = 0                    Empty REMOTE bins  = 0
Control functions in progress:
Backup         = N  Restore        = N
Verify         = N  Expiration     = N
Report Extract = N  Disaster Store = N
VRS            = N  Synchronize    = N
Client/Server:
  host name  =
  IP address =
```

Note: When running EDGHSKP expiration processing or vital record processing, CATSYSID(*) might have been specified in the EDGRMMxx PARMLIB member and the CDS might not be synchronized with the catalogs. In this case, DFSMSrmm initiates catalog synchronization even if you have not specified it.

Look at the message data set. You can see all data sets that are updated in the DFSMSrmm CDS by this run. Example 9-26 shows an example.

Example 9-26 Message data set after catalog synchronization

```
EDG6001I INVENTORY MANAGEMENT STARTING ON 2010/284 AT 14:04:47 -
         PARAMETERS IN USE ARE
         DATEFORM(J),CATSYNCH,DATE(2010/284)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
         VRSEL(NEW)
         VRSJOBNAME(2)
         VRSMIN(1,INFO)
         VRSCHANGE(INFO)
         VRSDROP(PERCENT(10),INFO) VRSRETAIN(PERCENT(80),INFO)
                 EXPDTDROP(PERCENT(10),INFO)
         SMSTAPE(PURGE(ASIS) UPDATE(EXITS,SCRATCH,COMMAND))
         CATRETPD(6)
         UNCATALOG(Y)
         TPRACF(N)
         NOTIFY(N)
         SYSID(SC70)
         CATSYSID(*)
         RETAINBY(SET)
         MOVEBY(SET)
         GDG(CYCLEBY(GENERATION),DUPLICATE(BUMP))
EDG2234I DFSMSrmm CDS CATALOG STATUS UNKNOWN FOR SCHLUM.RMMDEMO.FILE4.VOL23
         VOLUM.RMMDEMO.FILE4.VOL23 VOLUME SC0010 FILE 3 SYNCHRONIZED TO
         CATALOG STATUS
EDG2234I CONT:-  YES
....
EDG2234I DFSMSrmm CDS CATALOG STATUS YES FOR WWZ038.GSD1997A.DUMP VOLUME
         SC0005 FILE 1 SYNCHRONIZED TO CATALOG STATUS NO
EDG2234I DFSMSrmm CDS CATALOG STATUS YES FOR WWZ038.GSD1997A.DUMP VOLUME
         SC0011 FILE 1 SYNCHRONIZED TO CATALOG STATUS NO
....
EDG2234I DFSMSrmm CDS CATALOG STATUS NO FOR WWZ038.SAVE.ZOS120.DU01A426
         VOLUME Q17062 FILE 1 SYNCHRONIZED TO CATALOG STATUS YES
EDG2234I DFSMSrmm CDS CATALOG STATUS NO FOR WWZ038.SAVE.ZOS120.DU01A483
         VOLUME Q17030 FILE 1 SYNCHRONIZED TO CATALOG STATUS YES
EDG2307I INVENTORY MANAGEMENT TASK CATSYNCH COMPLETED SUCCESSFULLY
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

9.4.5 Enabling the journal data set (optional)

When the EDGHSKP run is complete, activate the journal data set again. To activate the journal data set, follow these steps:

1. Restart DFSMSrmm with the EDGRMMxx PARMLIB member by using the command shown in Figure 9-28.

```
F DFRMM,M=xx
```

Figure 9-28 Restart DFSMSrmm subsystem

2. Back up the DFSMSrmm CDS and the journal data set, with the EDGHSKP PARM='ACKUP' utility. The DFSMSrmm journal data set is now active, nullified, and synchronized with the DFSMSrmm CDS after the backup. You see the messages shown in Figure 9-29.

```
EDG2121I DFSMSrmm JOURNAL FILE IS NOW CLEARED
EDG2121I DFSMSrmm JOURNAL FILE IS NOW ENABLED
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

Figure 9-29 DFSMSrmm journal enable messages

Note: If the journal was previously used, you must delete it and allocate a new one, because the CDS and JOURNAL control information no longer match.

9.4.6 Housekeeping considerations

Figure 9-30 on page 356 shows the correct sequence in which to run your daily housekeeping jobs so that all components are updated correctly and always are synchronized. The examples show only the most important parameters.

You can run the first and last jobs on any of the systems. However, we suggest that you run these jobs on the server (SYSG) or on the standard system (SYSH) on sysplex 2 that has direct access to the DFSMSrmm control data set. The catalog synchronization, as part of the expiration processing, must run in each plex separately. This way, you run the EDGHSKP expiration processing on either SYSA or SYSB (sysplex 1), on either system SYSC or SYSD (non-plex 1), and on either system SYSG or SYSH (sysplex 2). The CATSYSID operand in the EDGRMMxx PARMLIB member must be present on each system and must include only system IDs that have access to the available catalogs.

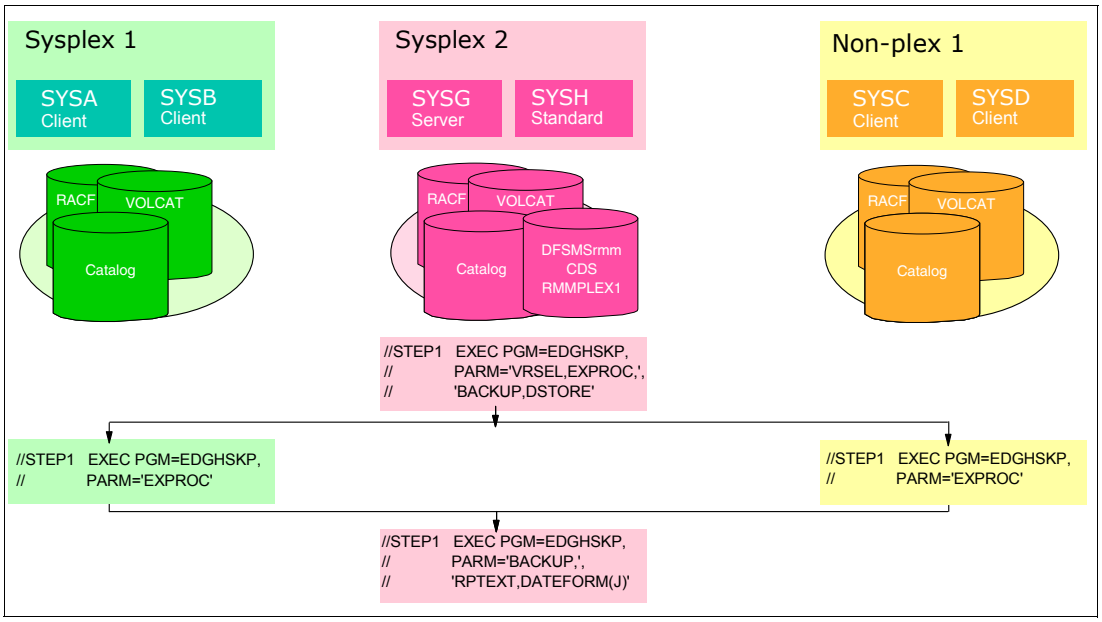


Figure 9-30 DFSMSrmm housekeeping when catalogs are not shared

Run inventory management functions on any one system at a time. However, if you run expiration processing to return volumes to scratch status, DFSMSrmm only processes a subset of the volumes.

You must run EDGHSKP expiration processing on multiple systems if you specify the EDGRMMxx PARMLIB OPTION UNCATALOG(YIS). Run expiration processing once for each set of user catalogs on a system with access to those user catalogs.

9.4.7 Catalog re-synchronization with a shared catalog environment

To request catalog re-synchronization, repeat the steps between 9.4.2, “Clearing the catalog synchronization” on page 350 and 9.4.5, “Enabling the journal data set (optional)” on page 355.

Note: Rerun catalog synchronization if you have manually added volumes in MASTER or USER status with data set information, or if you are changing your catalog environment (MERGECAT, IMPORT, and EXPORT).

10

Enabling ISPF Data Set List (DSLIST) support

In this chapter, we describe how to enable Interactive System Productivity Facility (ISPF) Data Set List support. The following topics are covered:

- Implementation steps
- Use the ISPF Configuration Utility
- Using the ISPF Data Set List Utility support
- Move the ISPCFIGU module to the SISPLPA library

10.1 Implementation steps

The ISPF Data Set List support can be implemented for a single user or it can be an installation-wide setting. Follow the steps to implement ISPF Data Set List support for a single user:

1. Create a new PDS with VB 255 to store the ISPF default settings.
2. Under ISPF, type `TSO ISPCCONF` on any command line.
3. Select Option 1 Create/Modify Settings and Regenerate Keyword File on the ISPF Configuration Utility primary panel and add the name to the keyword file dsname field you have allocated before, and add a member name. You see the "`Keyword file loaded`" message.
4. Select Option 3 PDF Exits and Other PDF Settings in the ISPF Configuration Utility primary panel.
5. Scroll to the correct place and add '/' to Enable RM/Tape Commands.
6. Exit the dialog by pressing the PF3 key twice. You see the "`Keyword file saved`" message.
7. Select Option 4 Build Configuration Table Load Module.
8. Specify any library in your ISPLLIB concatenation as the "Output Configuration Table Load Module Data Set".
9. Press Enter to load the module that is built.
10. Exit ISPF. Ensure that the specified load library is ISPLLIB concatenated.
11. Start ISPF. It works for a single user, but not for all users.

To implement the ISPF Data Set List support for all users, you can follow the steps as described in steps 1- 11 directly above, but the configuration table load module must be stored in the SISPLPA library. To load the module to the link pack area (LPA), you must IPL your system. For detailed information, see 10.4, "Move the ISPCFIGU module to the SISPLPA library" on page 367.

10.2 Use the ISPF Configuration Utility

Use the ISPF configuration table to change site-wide defaults and to indicate that installation exit routines are provided for some of the ISPF functions. The ISPF functions that allow installation-written exit routines are data set allocation, the print utility, data set compression, the Data Set List utility, member list filter, and data set name change. ISPF checks the configuration table to determine, first, whether exit routines are provided, and second, whether those routines are programs or CLISTs. If you specify both a CLIST and a program, ISPF uses the program.

Enter the TSO ISPCCONF command to start the ISPF Configuration Utility as shown in Figure 10-1 on page 359. You can choose the command in the ISPF command shell or on any available command line.

```
  Menu  Utilities  Compilers  Options  Status  Help
 ___________________________________________________________________________________
                              z/OS Primary Option Menu
Option ===> TSO ISPCCONF

0  Settings      Terminal and user parameters           User ID . : SCHLUM
1  View          Display source data or listings        Time. . . : 18:40
2  Edit          Create or change source data           Terminal. : 3278
3  Utilities     Perform utility functions              Screen. . : 1
4  Foreground    Interactive language processing        Language. : ENGLISH
5  Batch         Submit job for language processing     Appl ID . : PDF
6  Command       Enter TSO or Workstation commands      TSO logon : IKJACCT
7  Dialog Test   Perform dialog testing                 TSO prefix: MHLRES5
9  IBM Products  IBM program development products       System ID : SC70
10 SCLM          SW Configuration Library Manager       MVS acct. : ACCNT#
11 Workplace     ISPF Object/Action Workplace           Release . : ISPF 5.9
12 z/OS System   z/OS system programmer applications
13 z/OS User     z/OS user applications

     Enter X to Terminate using log/list defaults
```

Figure 10-1 Start the ISPF conversion utility

Figure 10-2 shows how you can select Option 1 Create/Modify Settings and Regenerate Keyword File after the ISPF conversion utility is up and running.

```
                          ISPF Configuration Utility                Enter option
 ___________________________________________________________________________________
 Option ===> 1

 1  Create/Modify Settings and Regenerate Keyword File
 2  Edit Keyword File Configuration Table
 3  Verify Keyword Table Contents
 4  Build Configuration Table Load Module
 5  Convert Assembler Configuration Table to Keyword File
 6  Build SMP/E USERMOD


 Keyword File Data Set
   Data Set . . . 'RMM.ADDONS.CEXECVB'
   Member . . . . SETTING

 Configuration Table Assembler Source Data Set
   Data Set . . .
   Member . . . .

   Output File Content for Keyword File
   2  1. Include only non-default values
      2. Include defaults as comments
      3. Include all values


 Current Configuration Table
   Keyword File : not available
   Identifier . : ISPCFIGU             Level  . . . : 480R8001
   Compile Date : 2003/04/04           Compile Time :
```

Figure 10-2 Create/Modify Settings and Regenerate Keyword File selection

After the keyword file is loaded, select Option 3 PDF Exits and Other PDF Settings as shown in Figure 10-3.

```
                        Create/Modify ISPF Configuration      Keyword file loaded
 ____________________________________________________________________________
 Option ===>  3

 General ISPF Settings                 System Profile (ISPSPROF) Settings
 1  Editor Settings                    6  Log and List Defaults
 2  Edit/View/Browse VSAM Settings     7  Terminal and User Defaults
 3  PDF Exits and Other PDF Settings   8  Workstation Defaults
 4  ISPF Site-wide Defaults            9  Workstation Download Defaults
 5  ISPDFLTS, CUA Colors, and Other
    DM Settings


 Output Keyword File
   Data Set . . . 'RMM.ADDONS.CEXECVB'
   Member . . . . SETTING


 Instructions:
   Enter option to change configuration settings,
   END or EXIT command to generate keyword file, or
   CANCEL command to exit without keyword file generation
```

Figure 10-3 PDF Exits and Other PDF Settings selection

The Modify PDF Configuration Settings menu is displayed in a scrollable panel. Scroll down until you can see the DSLIST Removable Media Settings as shown in Figure 10-4 on page 361.

```
                        Modify PDF Configuration Settings
 ____________________________________________________________________________
 Command ===>
                                                                   More:   -
      2. Use IEBCOPY for PDSEs only

   When to use COPY or COPYMOD
   2  1. Use COPY if the target block size is equal to or greater than the
         source block size, COPYMOD otherwise
      2. Use COPY if the target block size is equal to the source block size,
         COPYMOD otherwise
      3. Always use COPYMOD

 DSLIST Removable Media Settings
   Enter "/" to select option
   /  Enable RM/Tape Commands

   RM/Tape Command  . . %EDGRPD34
   Command APPLID . . . EDG

 Other PDF Settings
   Default PDF Unit . . . . . . . . . . . . SYSALLDA
   Volume for Migrated Data Sets  . . . . . MIGRAT
   Delete Command for Migrated Data Sets    HDELETE
   Allowed Allocation Units . . . . . . . . ANY
   Maximum IEBCOPY Return Code  . . . . . . 0
   Pathname Substitution Character  . . . . Ü

   Enter "/" to select option
      Allocate Before Uncatalog
   /  Verify Expiration Dates
   /  Use SuperC Program Interface
      Monitor Edit Macro Commands via the Activity Monitoring Exit
   /  Allow SUBMIT from Browse
   /  Allow SUBMIT from View
   /  Warn when rename target could be a GDG
   /  Default Edit/Browse/View member list from Option 3.4
   /  Enable View
      Use Panel ISRTSOA in Option 6
      Print using ICF
      Disallow wildcards in the high level qualifier for Data Set List
      Disable all ENQ displays
   /  Fail on LMF lock requests
```

Figure 10-4 Modify PDF Configuration Settings

After you complete your updates, you must verify your new keyword table using option 3 “Verify Keyword Table Contents” as shown in Figure 10-5 on page 362. Then, you must build a new configuration table using option 4 “Build Configuration Table Load Module” as shown in Figure 10-5 on page 362.

```
                         ISPF Configuration Utility
 ____________________________________________________________________________
 Option ===> 4

 1  Create/Modify Settings and Regenerate Keyword File
 2  Edit Keyword File Configuration Table
 3  Verify Keyword Table Contents
 4  Build Configuration Table Load Module
 5  Convert Assembler Configuration Table to Keyword File
 6  Build SMP/E USERMOD


 Keyword File Data Set
   Data Set . . . 'RMM.ADDONS.CEXECVB'
   Member . . . . SETTING

 Configuration Table Assembler Source Data Set
   Data Set . . .
   Member . . . .

   Output File Content for Keyword File
   2  1. Include only non-default values
      2. Include defaults as comments
      3. Include all values


 Current Configuration Table
   Keyword File : not available
   Identifier . : ISPCFIGU              Level  . . . : 480R8001
  Compile Date : 2003/04/04      Compile Time :
```

Figure 10-5 Build Configuration Table Load Module

A small pop-up window is displayed to specify the input keyword file data set, member, and the output configuration table load module data set. See Figure 10-6.

```
                    Build Configuration Table Load Module
  Command ===>

  Input Keyword File Data set
    Data Set . . . 'RMM.ADDONS.CEXECVB'
    Member . . . . SETTING

  Output Configuration Table Load Module Data Set
    Data Set . . . 'SCHLUM.ISPF.LLIB'

  Optional fields (leave blank for ISPF to use defaults)
    Object data set  . . .
    Configuration member             (Defaults to ISPCFIGU)
    VSAM member  . . . . .           (Defaults to ISPCFIGV)
```

Figure 10-6 Build Configuration Table Load Module pop-up window

If your target load library is a concatenated library of your LINKLIST, you must refresh the LINKLIST as shown in Example 10-1 on page 363.

Example 10-1 using the LLA refresh command

```
F LLA,REFRESH
```

Figure 10-7 shows the successful refresh of your linklist.

```
CSV210I LIBRARY LOOKASIDE REFRESHED
```

Figure 10-7 Successful refresh of linklist

After you create the new configuration table load module, ensure that the specified load library is concatenated to your ISPLLIB, exit ISPF, and start ISPF again.

Important: If the Data Facility System Managed Storage removable media manager (DFSMSrmm) REXX exec library, normally called SYS1.SEDGEXE1, is not concatenated in your logon procedure, you must move the two members, EDGRMLIB and EDGRPD34, to a concatenated REXX exec library and update member EDGRPD34 to specify that the RMM ISPF environment is allocated:

```
UseIspfLibdef = true            /* <<   true or false               */
```

10.3 Using the ISPF Data Set List Utility support

After you have enabled the ISPF Data Set List (DSLIST) support and restarted your ISPF session, you can use this support. The line commands supported by DFSMSrmm are I, S, M, and D:

I Displays a search results list showing all data sets in the multivolume set for the selected data set.

S Displays the individual data set details. DFSMSrmm determines the first file on the selected volume that matches the selected data set. If other data sets of the same name exist on the volume, the wrong details might be displayed. In that case, use the M line command and then the DFSMSrmm I line command from that results list.

M Displays a search results list showing all data sets defined to DFSMSrmm that match the selected data set name.

D Releases the volume. If the volume is part of a multivolume set, an option is shown to release all volumes in the set.

Figure 10-8 on page 364 shows a display of individual data set details after you enter the `S` line command.

```
  Menu  Options  View  Utilities  Compilers  Help
 ___________________________________________________________________________
DSLIST - Data Sets Matching SCHLUM.LI*                            Row 1 of 5
Command ===>                                                 Scroll ===> PAGE

Command - Enter "/" to select action                 Message          Volume
-------------------------------------------------------------------------------
         SCHLUM.LIBDUMP                                                SBOX03
         SCHLUM.LIBDUMP.BIN.XMIT                                       SBOXFG
         SCHLUM.LINEITEM.PUNCH                                         SBOX81
 I       SCHLUM.LINEITEM.UNLOAD                      Info - I          TST026
         SCHLUM.LISTDEF.INPUT                                          SBOX09
***************************** End of Data Set list ****************************
```

Figure 10-8 Displaying individual data set details

The result is shown in Figure 10-9 on page 365.

```
  Panel  Help
 ------------------------------------------------------------------------------
                          DFSMSrmm Data Set Details
 Command ===>

 Data set name . . : 'SCHLUM.LINEITEM.UNLOAD'
 Volume serial . . : TST026      Physical file sequence number . . : 1
 Owner . . . . . . : SCHLUM      Data set sequence number  . . . . : 1
                                                                     More:
 Job name  . . . . : SCHLUMR3U
 Step name . . . . : STEP010                Record format . . . . : VB
 Program name  . . : DSNUTILB               Block size  . . . . . : 32760
 DD name . . . . . : SYSREC                 Logical record length : 151
 Create date . . . : 2003/300    YYYY/DDD   Block count . . . . . : 36140
 Create time . . . : 14:51:16               Total block count . . : 36140
 Expiration date . : 2003/300    YYYY/DDD   Percent of volume . . : 2
 Original  . . . . :             YYYY/DDD   Device number . . . . : 0B92
 System id . . . . : SC63

 Last job name . . : PAOLOR3T               Last DD name  . . . . : TAPEVOL
 Last step name  . : STEP010                Last device number  . : 0B92
 Last program name : IDCAMS
 Date last read  . : 2003/302               VRS management value  :
 Date last written : 2003/300               Management class  . . : MCDB22
                                            Data class  . . . . . :
 Retention date  . :                        Storage class . . . . : SCLIB1
 VRS retained  . . : NO                     Storage group . . . . : SGLIB1

 Security name . . :
 Classification  . :

 Primary VRS details:
  VRS name . . . :
  Job name . . . :                            VRS type  . . . . . :
  Subchain name  :                            Subchain start date :

 Secondary VRS details:
  Value or class :
  Job name . . . :
  Subchain name  :                            Subchain start date :

 Catalog status  . : YES
```

Figure 10-9 Result of displaying individual data set details

Figure 10-10 on page 366 shows all data sets that are defined to DFSMSrmm that match the selected data set name using the M line command.

```
  Panel  Help  Scroll
 ____________________________________________________________________________
                DFSMSrmm Data Sets (Page 1 of 2)              Row 1 to 1 of 1
Command ===>                                                 Scroll ===> PAGE


Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
                                                 Volume          File
S  Data set name                                 serial Owner    seq
-- ------------------------------------------ ------ -------- -----
   SCHLUM.LINEITEM.UNLOAD                        TST026  SCHLUM  PAOLOR3
****************************** Bottom of data ********************************
```

Figure 10-10 Result if you select function M

Use the D line command to release the volume as shown on the data set. If you select this function, DFSMSrmm shows you the Confirm Volume Release panel as shown in Figure 10-11.

```
  Panel  Help
 ____________________________________________________________________________
                      DFSMSrmm Confirm Volume Release
Command ===>

Volume . . . . . . : TST026       Location . . . . . . . . : LIB1
  VOL1 volser  . . :              In container . . . . . . :
Volume type  . . . : PHYSICAL     Rack number  . . . . . . : TST026
Media name . . . . : MEDIA3       Expiration date  . . . . : 2003/365

                                  Original expiration date :
Status . . . . . . : USER
Description  . . . :
Data set name  . . : 'SCHLUM.LINEITEM.UNLOAD'
                                  Release actions:
Media type . . . . : HPCT           Return to SCRATCH pool : NO
Label  . . . . . . : SL             Replace volume . . . . : NO
Density  . . . . . : IDRC           Return to owner  . . . : YES
Recording format . : 128TRACK       Initialize volume  . . : NO
Compaction . . . . : YES            Erase volume . . . . . : NO
Attributes . . . . : NONE           Notify owner . . . . . : NO
Availability . : PENDING RELEASE    Expiry date ignore . . : NO
                                    Scratch immediate  . . : NO
Press ENTER to NOFORCE volume, or END command to CANCEL.
```

Figure 10-11 Result if you select the function to release the volume

Figure 10-12 on page 367 shows a search result if you have used the I line command that includes all volumes that reside on the same tape or all data sets that reside on the multi-volume set.

```
  Panel  Help  Scroll
 ____________________________________________________________________________
                 DFSMSrmm Data Sets (Page 1 of 2)            Row 1 to 10 of 10
Command ===>                                                  Scroll ===> PAGE


Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
                                                 Volume          File
S  Data set name                                 serial Owner    seq
-- ------------------------------------------- ------ -------- -----
   SCHLUM.LINEITEM.UNLOAD                        TST026 MHLRES5  1
   SCHLUM.TESTSTAC.TESTMF02                      TST026 MHLRES5  2
   SCHLUM.TESTSTAC.TESTMF03                      TST026 MHLRES5  3
   SCHLUM.TESTSTAC.TESTMF04                      TST026 MHLRES5  4
   SCHLUM.TESTSTAC.TESTMF05                      TST026 MHLRES5  5
   SCHLUM.TESTSTAC.TESTMF06                      TST026 MHLRES5  6
   SCHLUM.TESTSTAC.TESTMF07                      TST026 MHLRES5  7
   SCHLUM.TESTSTAC.TESTMF08                      TST026 MHLRES5  8
   SCHLUM.TESTSTAC.TESTMF09                      TST026 MHLRES5  9
   SCHLUM.TESTSTAC.TESTMF10                      TST026 MHLRES5  10
****************************** Bottom of data ********************************
```

Figure 10-12 SEARCH volume result

10.4 Move the ISPCFIGU module to the SISPLPA library

After you have installed and verified ISPF, you can enhance its performance by adding the LPA-eligible load modules (in the SISPLPA library) to the LPA list in an LPALSTxx member of PARMLIB. Add those load modules not eligible for LPA (in the SISPLOAD library) to the link list in an LNKLSTxx member of parmlib. For information about adding data sets to the Link and LPA lists, see *z/OS MVS Initialization and Tuning Reference,* SA22-7592. You can then remove these data sets from the STEPLIB in your TSO LOGON procedure. After adding SISPLPA to LPALST and SISPLOAD to LNKLST, specify CLPA as an initial program load (IPL) parameter to force the SISPLPA modules into the link pack area and to add SISPLOAD as a system link library.

Finally, after you test everything, you can move the ISPCFIGU member to your ISP. You must IPL your system again or refresh your linklist as shown in Example 10-2. Then, you must log on to your system again before you can use this new service.

Example 10-2 Using the LLA refresh command

```
F LLA,REFRESH
```

Figure 10-13 shows the successful refresh of your linklist.

```
CSV210I LIBRARY LOOKASIDE REFRESHED
```

Figure 10-13 Result of the LLA REFRESH

11

Policy management

This chapter describes how Data Facility System Managed Storage removable media manager (DFSMSrmm) provides policy management for movement and retention at the data set level. Every tape data set can have a policy, and each policy can specify movement as well as retention. The retention and movement policies that you define to DFSMSrmm are known as *vital record specifications* (VRSs). You use them to indicate how long and where you want to keep data sets or volumes. You also use them to define how volumes are to be moved among the libraries that DFSMSrmm supports, and the storage locations defined for vital records and disaster recovery purposes.

This chapter contains the following information:

- VRS-related parmlib options
- VRS types
- Specifying the VRS parameters
- Chaining VRSs
- VRSEL GDG option
- DFSMS ACS support
- Modifying VRS
- Deleting VRS
- DFSMSrmm VRS policy management simplification
- Trial run
- VRS examples
- Complex VRS chain with three subchains
- VRS hints and tips

By the end of this chapter, you will understand the different VRS types and how to create them. Sample VRS definitions are provided for better understanding.

11.1 VRS-related parmlib options

For DFSMSrmm VRS processing, we have several PARMLIB options that can be used for tailoring your environment:

CATRETPD Specifies the number of hours from creation that a data set needs to be retained if it has not been cataloged and matches a VRS with the WHILECATALOG operand.

CATSYSID Specify CATSYSID to enable DFSMSrmm catalog synchronization. Use the CATSYSID operand to identify the user catalogs you want tracked when you run the DFSMSrmm EDGHSKP utility with the CATSYNCH operand to exploit catalog status tracking.

GDG Use the GDG option to specify how generation data groups (GDGs) are handled for cycle retention by VRSEL processing.

MOVEBY Specify the MOVEBY operand to move volumes that are retained by DFSMSrmm VRSs as a set of volumes or as individual volumes.

RETAINBY Use the RETAINBY option to specify whether DFSMSrmm retains multivolume sets as a set or as individual volumes.

RETENTIONMETHOD Use this operand to set the system-wide retention method for new tape volume sets that are created during Open/Close/End of Volume (O/C/EOV) processing, and for tape volumes added to the DFSMSrmm control data set (CDS).

REUSEBIN Specifies control for how DFSMSrmm reuses bins when a volume is moving from a bin.

VRSCHANGE Use VRSCHANGE to specify the action that DFSMSrmm needs to take during inventory management if you have made any changes to VRSs.

VRSDROP Use VRSDROP to specify a maximum number or percentage of existing VRS-retained volumes that can be dropped from vital records retention and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of VRS-retained physical and logical volumes at the start of vital record selection processing and the number of these dropped by vital record processing. When you specify COUNT, this is an absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing. When you specify PERCENT, this is the maximum percentage of the existing VRS-retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

VRSJOBNAME Use VRSJOBNAME to select how DFSMSrmm uses a job name in a VRS to match a data set to a movement and retention policy.

VRSMIN Use VRSMIN to specify a minimum number of VRSs required by inventory management vital record processing, and the action to be taken by DFSMSrmm when the minimum number of VRSs is not defined.

VRSEL Use VRSEL to control how DFSMSrmm inventory management vital record processing uses retention and movement information that is defined in VRSs.

Restriction: The **VRSEL(OLD)** option is no longer supported. **VRSEL(NEW)** is the only valid value for the VRSEL option.

VRSJOBNAME Use VRSJOBNAME to select how DFSMSrmm uses a job name in a VRS to match a data set to a movement and retention policy. DFSMSrmm records the data set name and job name for new tape files you create. You can define the name of the job that created a data set by using the RMM ADDDATASET subcommand or the CHANGEDATASET subcommand for existing data sets. You can specify a job name and data set name in a VRS so that data sets that match the data set name and job name are moved and retained by the policy that is defined in the VRS.

For detailed information, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

11.1.1 CATRETPD

DFSMSrmm retains the data set for the catalog retention period if the data set has never been cataloged. DFSMSrmm does not retain the data set if DFSMSrmm detected that the data set was cataloged, and then uncataloged, during the catalog retention period.

You have to specify a higher catalog retention value than the default if you have a job creating data sets on tape with a long running time, and the data sets expand a volume, or the job creates more than one data set. If the inventory management process is starting while the job is running, you cannot check whether a data set is cataloged, because the data sets are not cataloged until the job is finished. If you have not defined a catalog retention greater than the run time of your jobs, you can lose some data sets because DFSMSrmm releases them before the job is finished.

Figure 11-1 illustrates the following situation: The default retention in the parmlib member is set to zero RETPD(0). A data set name vital record definition that matches all data sets exists, but while cataloged only. You have started a long running job at 7:00 PM, creating one or more data sets expanding a single volume. You have not specified a retention parameter in your JCL. Early in the morning at 8:00 AM, 13 hours after your job has started, the daily EDGHSKP with the inventory management function is starting, and checks all data sets for matching VRSs.

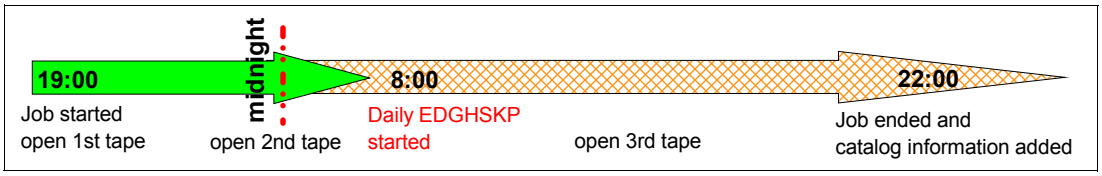


Figure 11-1 How catalog retention works

If you have not specified a CATRETPD parameter in your parmlib, the default value 12 hours is used and the first volume is released, because the data set on the volume has no retention, and is not cataloged so the VRS definition does not match.

If you have specified a CATRETPD parameter in your parmlib with the value 24 hours, the data set on the first volume is protected by this definition, and no more checks are done. The volume is not released.

11.1.2 GDG option

Use the GDG option to specify how generation data groups (GDGs) are handled for cycle retention by VRSEL processing. Cycle retention includes both the CYCLES and the BYDAYSCYCLE retention types. The correct sequence for determining the retention can be either by using the generation number or the creation order. You can also specify how duplicate generations (generation data sets) are handled and have the flexibility to include or exclude duplicate generations from the cycles count as required by your application processing.

The GDG option has two operands, CYCLEBY and DUPLICATE.

CYCLEBY

Use the CYCLEBY operand to specify whether retention is to be based on generation number or on creation date:

CYCLEBY(GENERATION)

Specifies that retention is to be based on the generation number. DFSMSrmm determines the generation number by applying a similar algorithm to that used by Catalog processing. Both the creation order and the generation number from the data set name are considered, allowing wraps in generation number to be correctly handled.

CYCLEBY(GENERATION) is the default.

CYCLEBY(CRDATE) Specifies that only the creation sequence is to be used to determine the retention.

DUPLICATE

Use the DUPLICATE operand to specify how VRSEL processing handles duplicate generations. You can specify one of the following options:

- Count duplicate generations
- Keep duplicate generations, but do not count them
- Bump duplicate generations from the current subchain
- Drop duplicate generations from VRS retention

Duplicate generations are determined within a single VRS subchain and only if the generation and the generation it duplicates are not already dropped for another reason. For GENERATION-based cycles, the duplicate generations are determined in generation (data set name), then creation order. For creation date-based cycles, the duplicate generations are determined only in creation order, so they can be detected only when they are created consecutively.

Duplicates are processed within the context of the matching VRS chain or subchain, depending on the DUPLICATE option. DROP is within the context of the VRS chain and all others are within the context of the subchain. A *duplicate generation* is considered a duplicate for cycle retention only if the generation it duplicates is retained by the current VRS subchain.

DUPLICATE(BUMP) Specifies that a duplicate generation is to be bumped by the current VRS subchain and considered for retention by a subsequent VRS subchain.

DUPLICATE(BUMP) is the default.

DUPLICATE(COUNT) Specifies that a duplicate generation is to be treated like a non-duplicate generation regarding CYCLE and BYDAYSCYCLE processing.

DUPLICATE(DROP) Specifies that a duplicate generation is to be dropped from VRS retention without further consideration.

DUPLICATE(KEEP) Specifies that a duplicate generation is considered as either the same CYCLE or the same BYDAYSCYCLE, depending on the VRS retention type and regardless of the duplicate generation creation date.

Before we look at different handling options that can be specified, the circumstances are listed under which VRSEL applies the GDG(DUPLICATE()) operand.

1. Same generation number as previous.

   Consider GDG(CYCLEBY(CRDATE)). A higher generation might have already been created before the same generation has been created again (such as the example shown in Figure 11-2).

2. Same matching VRS as previous (same subchain).

   Data sets with the same name and generation number might match a different VRS for the following reasons:

   – DELETE VRS, OPEN VRS, and ABEND VRS
   – VRS with Management Class
   – VRS with JOBNAME()

   DUPLICATE processing only applies within a single VRS subchain.

3. The VRS retention type is CYCLES or BYDAYSCYCLE.
4. Other retention criteria are met (for example, UNTILEXPIRED).

Creation Sequence	Data Set Name	Matching VRS	Processing sequence	Duplicate
	A.B.G0003V00	DSN('A.B.*') CYCLES		YES
	A.B.G0003V00	DSN('DELETE') DAYS		NO (3)
	A.B.G0003V00	DSN('A.B.*') CYCLES		NO (1)
	A.B.G0004V00	DSN('A.B.*') CYCLES		NO (1)
	A.B.G0003V00	DSN('A.B.*') CYCLES		NO (1)
	A.B.G0003V00	DSN('DELETE') DAYS		NO (2,3)

Figure 11-2 Process data set as Duplicate Generation

The table in Figure 11-3 on page 374 shows the different results of the DUPLICATE operands BUMP, DROP, KEEP, and COUNT.

Assume that you have three generation data sets in a cycle group with the "red" generation 3 being processed as a "duplicate".

The matching VRS has two subchains. The first subchain keeps two cycles, and the second one has a DAYS retention:

BUMP The duplicate does not get retained by the CYCLES subchain, but the next subchain retains it (assuming that the retention criteria are met).

DROP The duplicate is dropped from the whole VRS chain.

KEEP The duplicate gets retained by the CYCLES subchain as the same cycle as it's original, so that three data sets are kept with COUNT(2).

COUNT The duplicate also gets retained by the CYCLES subchain. However, because it has its own cycle, generation 2 cannot be kept in this subchain anymore.

DUPLICATE processing works in a similar manner to CYCLES and BYDAYSCYCLE retention.

For BYDAYSCYCLE, if the duplicate is created on the same day as its original, KEEP and COUNT make no difference because BYDAYSCYCLE data sets that are created on the same day are treated as one cycle.

VRS Subchains	GDG DUPLICATE(...)			
	BUMP	DROP	KEEP	COUNT
CYCLES COUNT(2)	A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0003V00
DAYS COUNT(100)	A.B.G0003V00			A.B.G0002V00

Figure 11-3 Different results of the DUPLICATE operands BUMP, DROP, KEEP, and COUNT

11.1.3 MOVEBY

You use this parameter to move volumes that are retained by DFSMSrmm VRSs as a set of volumes or as individual volumes:

- **VOLUME**: Specify VOLUME when you want to move each volume individually.
- **SET**: Specify SET if you want DFSMSrmm to move a multivolume set as an aggregate. When you retain by SET, if any volume in a set is to be moved by a VRS, all volumes in the set are moved to the same location.

Figure 11-4 on page 375 shows how DFSMSrmm works with the MOVEBY(SET) specification if you have a set of three volumes with three data sets on it, and two of the data sets need to be moved to different locations.

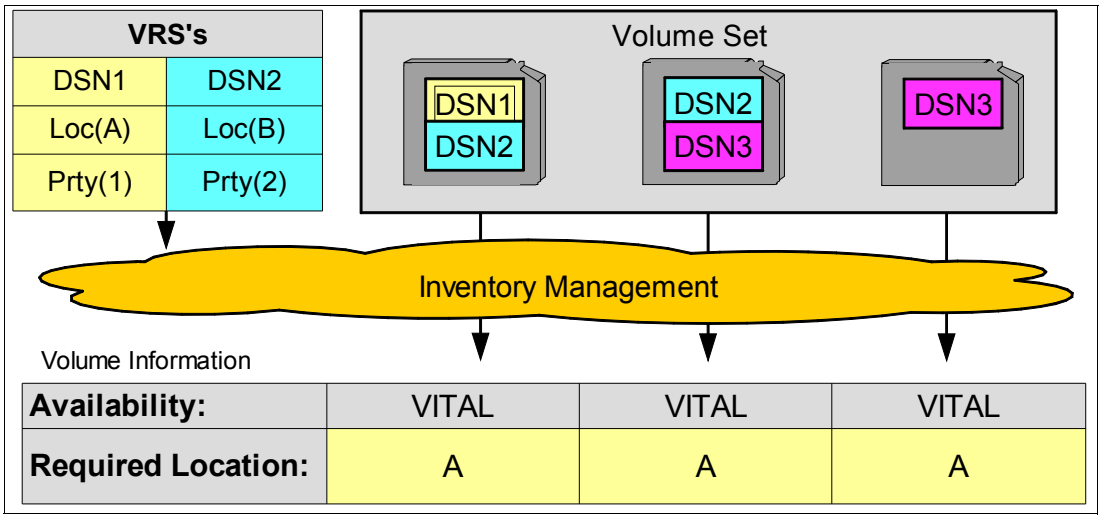


Figure 11-4 RETAINBY(SET) MOVEBY(SET)

The second volume is moved to location A because the VRS definition for data set A has a higher priority than the VRS definition for data set B. The third volume in the set is moved along with the other two volumes because it is part of the set.

11.1.4 RETAINBY

This operand allows you to work with the volume set as an aggregate:

- **VOLUME**: Specify VOLUME when you want to retain each volume individually.
- **SET**: Specify SET if you want DFSMSrmm to retain a multivolume set as an aggregate. When you retain by SET, if any volume in a set is retained by a VRS, all volumes in the set are retained as vital records.

  When a volume in a set is only retained because it is a member of a set, DFSMSrmm sets an indicator as shown in Figure 11-5.

```
Set retained . . . : YES
```

Figure 11-5 RETAINBY set indicator

Figure 11-6 shows how DFSMSrmm works with the RETAINBY(SET) specification if you have a set of three volumes with three data sets on it.

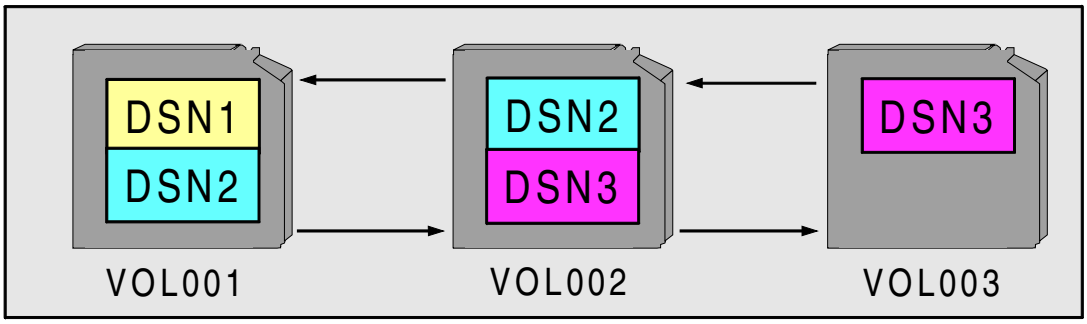


Figure 11-6 Volume set

Figure 11-7 shows VOL001 with EXPDT +10 days, VOL002 with EXPDT +20 days, and VOL003 with EXPDT -1 day. After the housekeeping, all the volumes are retained and VOL003 sets the Set Retained field to `YES`, because it is retained only by being a member of a set.

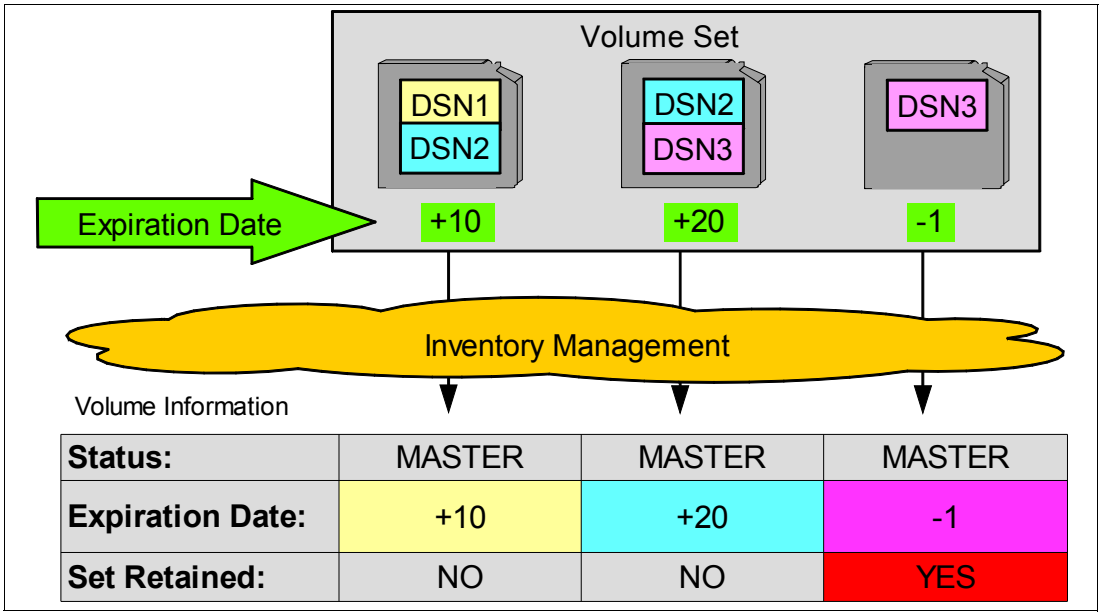


Figure 11-7 RETAINBY(SET) - EXPDT

If the volumes are retained by VRS, you see the result shown in Figure 11-8.

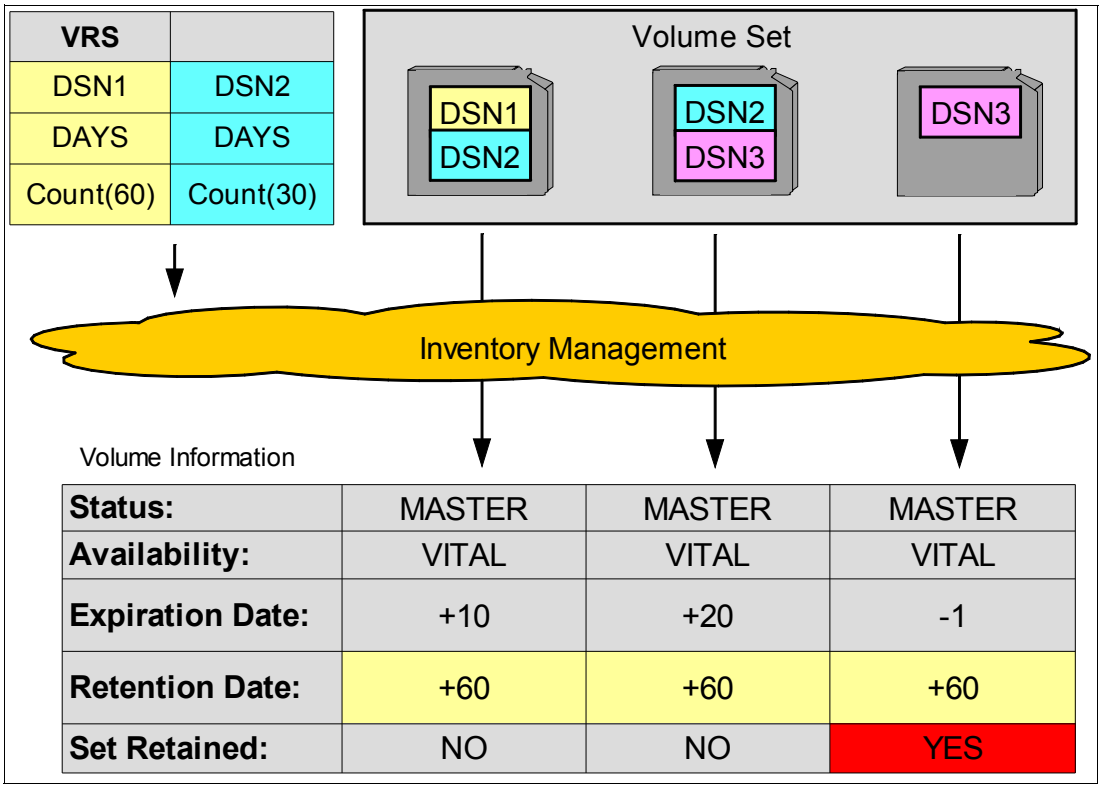


Figure 11-8 RETAINBY(SET) - VRS

11.1.5 REUSEBIN

Use this operand to control how DFSMSrmm reuses bins when a volume is moving from a bin.

CONFIRMMOVE

When a volume moves out of a bin, DFSMSrmm does not reuse this bin until the volume move has been confirmed. In this case, you have to define more bins, as shown in Figure 11-9.

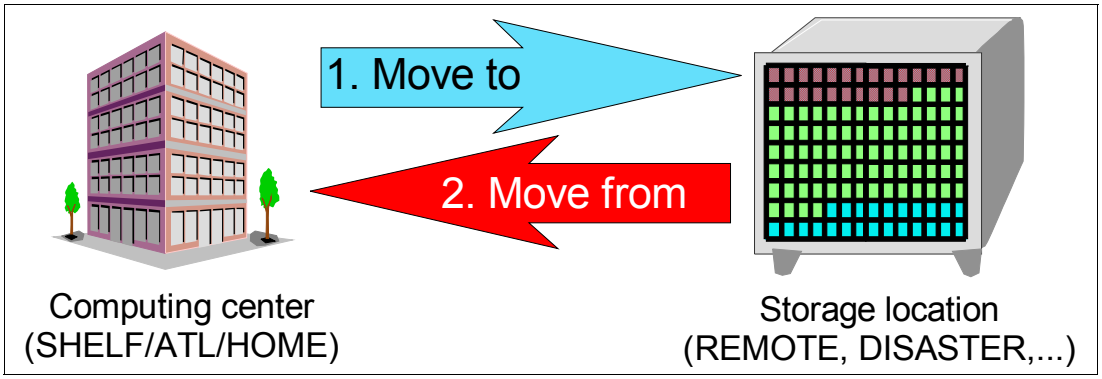


Figure 11-9 REUSE bin with CONFIRMMOVE operand

STARTMOVE

When a volume moves out of a bin, DFSMSrmm does not reuse this bin until the volume move has been confirmed.

A bin can be reused as soon as a volume starts moving out of a bin. Extended bin support must be enabled before you can use this operand. This parameter works only if you have enabled the extended bin support. For more information, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405. The bins are reused in the same processing run as shown in Figure 11-10.

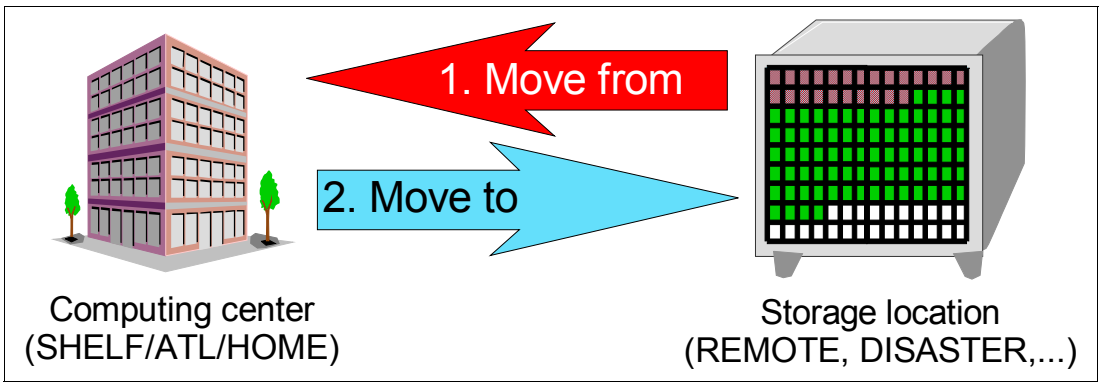


Figure 11-10 REUSE bin with STARTMOVE operand

11.1.6 VRSCHANGE

Specify the action for DFSMSrmm to take during inventory management if you have made any changes to VRSs since the last run:

- **INFO**: You see a message in the job log and no additional processing or actions are required when VRS changes occur.

- **VERIFY**: Any changes made to VRSs must be verified by running EDGHSKP vital record processing using the VERIFY parameter. DFSMSrmm must issue a return code zero before EDGHSKP vital record processing can be performed to update the control data set.

> **Note:** Check the ACTIVITY log of the vital record processing trial run, and check all changes made in this run before you start the lifecycle update.

11.1.7 VRSJOBNAME

Use this option to tell DFSMSrmm which of the two possible masks, data set name (DSNAME) or job name (JOBNAME) to check first:

1 If you specify one, the jobname or jobname mask is checked first, and the data set name or data set name mask is checked second.

2 If you specify two, the data set name or data set name mask is checked first, and the jobname or jobname mask is checked second.

As part of vital record processing, DFSMSrmm checks that data set name masks and job name masks used in VRSs are specified according to DFSMSrmm naming conventions.

DFSMSrmm compares the data set, job name, and volume information recorded in the control data set with information in the VRSs to determine which data sets and volumes to retain and the processing required. This includes any volumes with special JCL-specified expiration dates used by your installation.

DFSMSrmm builds a matching string with the data set and job name. The data set and job name position depends on the value of the VRSJOBNAME parameter. DFSMSrmm selects the most specific VRS.

For example, we define the VRSs as shown in Figure 11-11.

```
1. RMM ADDVRS DSN('ZE*.TEST.T¬¬¬¬R¬¬') WHILECATALOG
2. RMM ADDVRS DSN('*ERB.TEST.T¬¬¬¬R¬¬') DAYS COUNT(150)
3. RMM ADDVRS DSN('*ERB.TEST.T¬¬¬¬R¬¬') CYCLES COUNT(10)  JOBNAME('ZERB%%%')
```

Figure 11-11 VRS examples using wildcard characters

We also run job ZERB0A, which creates the data set ZERB.TEST.T0023R45.

Depending on the VRSJOBNAME parameter, we get different results.

Using the VRSJOBNAME(1) option, we have the following syntax string as shown in Figure 11-12.

```
ZERB0A!!ZERB.TEST.T00023R45
```

Figure 11-12 VRSJOBNAME(1)

In this case, VRS 3 in Figure 11-11 matches the string, because it matches first by job name and second by data set name.

Using VRSJOBNAME(2), we have the following syntax string as shown in Figure 11-13 on page 379.

```
ZERB.TEST.T00023R45!!ZERB0A
```

Figure 11-13 VRSJOBNAME(2)

In this case, VRS 1 in Figure 11-11 on page 378 matches the string, because it matches by the data set name having an alphanumeric character (Z) in the first position.

11.1.8 VRSMIN

Specify a minimum number of VRSs required by inventory management vital record processing, and the action to be taken by DFSMSrmm when the minimum number of VRSs is not defined. DFSMSrmm counts the VRSs used by vital record processing. DFSMSrmm does not count VRSs that are deleted during vital record processing.

FAIL Issues message EDG2229I to the MESSAGE file and stops inventory management processing. Processing ends with return code 8.

INFO Issues message EDG2229I to the MESSAGE file and processing continues.

WARN Issues message EDG2229I to the MESSAGE file and sets a minimum return code of 4. Processing continues.

Tip: After conversion, set the minimum number of VRSs that you get, and set the processing action to FAIL.

11.1.9 VRSEL

Use VRSEL to control how DFSMSrmm inventory management vital record processing uses retention and movement information that is defined in VRSs:

- **OLD**: This option is no longer supported.
- **NEW**: Specify NEW if you want DFSMSrmm to process the following information:
  - Retention information in NAME VRSs
  - Release options defined in VRSs
  - VRSs chained using the ANDVRS operand

The following example shows how VRSEL(NEW) works:

- We first define the following data set and management value VRSs as shown in Figure 11-14.

```
RMM ADDVRS DSNAME('YCJRES.TEST')              -
           DAYS COUNT(100)                    -
           LOCATION(REMOTE)
RMM ADDVRS DSNAME('CYCL004')                  -
           CYCLES COUNT(4) LOCATION(DISTANT)  -
           STORENUMBER(4)
```

Figure 11-14 Defining two data set name VRS definitions

- Figure 11-15 on page 380 shows several data sets that were created that have a management value of CYCL004 assigned.

```
(1) Data set name YCJRES.TEST creation date 120 days ago
(2) Data set name YCJRES.TEST creation date 119 days ago
(3) Data set name YCJRES.TEST creation date 118 days ago
(4) Data set name YCJRES.TEST creation date  90 days ago
(5) Data set name YCJRES.TEST creation date  89 days ago
(6) Data set name YCJRES.TEST creation date  88 days ago
```

Figure 11-15 Data set created using a management value of CYCL004

- When VRSEL(NEW) is in use, both VRSs are processed. Data set numbers (4), (5), and (6) are retained in location REMOTE by the data set VRS. Data set number (3) is retained in the DISTANT location by the name VRS. DFSMSrmm retains the data set (3) by the secondary VRS (MV VRS) after the primary VRS (data set VRS) does not retain the data set any more.

With VRSEL(NEW), both VRSs (primary and secondary) are processed in parallel on each housekeeping run. DFSMSrmm evaluates both VRSs and then determines which policy to apply.

When VRSEL(NEW) is in use, the volume expiration date is not used during data set name VRS UNTILEXPIRED processing when a data set matches both a primary VRS and a secondary VRS. For example, if a retention type of UNTILEXPIRED is defined in the primary data set name VRS, and in the secondary management value (name VRS) VRS, a retention type of DAYS with COUNT(30) is defined, and then the volume is retained by the secondary management value VRS. This definition retains the data set for 30 more days before being expired.

If UNTILEXPIRED is used in a secondary VRS, the volume expiration date is used to determine whether the retention condition is true.

There is no restriction on the definition of either the primary or secondary VRS; both can be VRS chains containing any of the possible ADDVRS operands.

11.1.10 VRSDROP

Use VRSDROP to specify a maximum number or percentage of existing VRS-retained volumes that can be dropped from vital records retention, and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of VRS-retained volumes at the start of vital record selection processing and the number of these dropped by vital record processing. When you specify COUNT, this is an absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing. When you specify PERCENT, this is the maximum percentage of the existing VRS-retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

Figure 11-16 on page 381 shows the correct use of the VRSDROP parameter option.

```
                    |—VRSDROP(PERCENT(10),INFO)————————————)—|
                    |                                        |
▶▶———OPTION————————————————————————————————————————————————————▶
                    |                                        |
                    |          |—PERCENT(—%age—)—|   |—INFO—|   |
                    |—VRSDROP(——————————————————,————————)—|
                    |          |—COUNT(— count—)—|   |—WARN—|
                                                     |—FAIL—|
                                                     |—OFF——|
```

Figure 11-16 EDGRMMnn OPTION operand VRSDROP

The following explanations relate to VRSDROP:

COUNT This is an absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing.

PERCENT This is the maximum percentage of the existing VRS-retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing. VRSDROP processing is intended to provide limited checking for volumes that are already VRS retained. It considers how many of these existing VRS-retained volumes are removed from VRS retention. Those volumes dropped from VRS retention can be set pending release, depending on VRS release options and the volume expiration date. They can also become EXPDT retained and on the next run of EXPROC, they are considered by EXPDTDROP limit processing. COUNT can be 0 - 2,147,483,647. PERCENT can be 0 to 100. The default is PERCENT(10).

action Specify an action to control the action DFSMSrmm takes during processing and when the value is exceeded. Action can be FAIL, INFO, OFF, or WARN.

- **FAIL** Issues messages EDG2242I and EDG2244I to the MESSAGE file. When the value is exceeded, DFSMSrmm stops inventory management processing before making CDS updates. In addition, message EDG2310I is issued. Processing ends with return code 12.
- **INFO** Issues messages EDG2242I and EDG2244I to the MESSAGE file and processing continues.
- **OFF** Processing of this function is turned off.
- **WARN** Issues messages EDG2242I and EDG2244I to the MESSAGE file. When the value is exceeded, DFSMSrmm sets a minimum return code of 4 and processing continues.

11.1.11 VRSRETAIN

Use VRSRETAIN to specify a minimum number or percentage of newly assigned volumes that are to be retained by vital records retention, and the action to be taken by DFSMSrmm. A newly assigned volume has a volume assignment date and time that are more recent than the run date and time of the previous VRSEL processing. DFSMSrmm counts the number of newly assigned volumes at the start of vital record selection processing and the number of these volumes that become VRS retained by vital record processing.

When you specify a count, this is an absolute minimum number of volumes that are to be retained by a single run of EDGHSKP VRSEL processing.

When you specify a percentage, this is the minimum percentage of the newly assigned volumes that are to be retained by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

Figure 11-17 shows the correct use of the VRSRETAIN parameter option.

```
                    |—VRSRETAIN(PERCENT(80),INFO)————————————————|
                    |                                            |
 ▸▸———OPTION————————————————————————————————————————————————————▸
                    |                                            |
                    |             |—PERCENT(—%age—)—|   |—INFO—|   |
                    |—VRSRETAIN(——————————————————,————————)——|
                                  |—COUNT(— count—)—|   |—WARN—|
                                                        |—FAIL—|
                                                        |—OFF——|
```

Figure 11-17 EDGRMMnn OPTION operand VRSRETAIN

The following definitions relate to VRSRETAIN:

COUNT This is an absolute minimum number of volumes that are to be retained by a single run of EDGHSKP VRSEL processing.
A newly assigned volume is one that has a volume assignment date and time that are higher (more recent) than the run date and time of the previous VRSEL processing and the volume is not VRS retained. DFSMSrmm counts the number of newly assigned physical and logical volumes at the start of vital record selection processing and the number of these volumes that become VRS retained by vital record processing.

PERCENT This is the minimum percentage of the newly assigned volumes to be retained by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing. VRSRETAIN processing is intended to provide limited checking for volumes containing newly created data sets that have not yet been processed by inventory management VRSEL. The data sets on the volumes have been created since the start of the last completed VRSEL run. It considers how many of these volumes become VRS retained during the new VRSEL run. Those volumes that become VRS retained will be considered by the VRSDROP limit checking in future VRSEL runs. Those volumes that are not retained by VRS can be set pending release, depending on VRS release options and the volume expiration date. They can also become EXPDT retained and on the next run of EXPROC, they are considered by EXPDTDROP limit processing. COUNT can be 0 - 2,147,483,647. PERCENT can be 0 - 100. The default is PERCENT(80).

action Specify an action to control the action DFSMSrmm takes during processing and when the value is exceeded:

FAIL Issues messages EDG2243I and EDG2245I to the MESSAGE file. When the value is exceeded, DFSMSrmm stops inventory management processing before making CDS updates. In addition, message EDG2310I is issued, and processing ends with return code 12.

INFO Issues messages EDG2243I and EDG2245I to the MESSAGE file and processing continues.

OFF Processing of this function is turned off.

WARN Issues messages EDG2243I and EDG2245I to the MESSAGE file. When the value is exceeded, DFSMSrmm sets a minimum return code of 4 and processing continues.

11.2 VRS types

DFSMSrmm uses policies to control the retention of data sets and volumes, and to determine a volume's required location. The retention and movement policies that are specified in the VRS are processed during DFSMSrmm inventory management (housekeeping).

You can define expiration and retention defaults for your installation by using the DFSMSrmm parmlib member EDGRMMxx.

There are three types of VRS:

- Data set
- Volume
- Name

To define a VRS, you can use DFSMSrmm ISPF panels or TSO subcommands. In the examples, we used TSO subcommands. To define a VRS by using panels, go to the DFSMSrmm Add Vital Record Specification panel, shown in Figure 11-18.

```
  Panel  Help
------------------------------------------------------------------------------
                   DFSMSrmm Add Vital Record Specification
Command ===>

Specify one of the following:
  Data set mask . . 'DELETEX'
       Job name mask . .          Add data set filter VRS . .    (Yes or No)

  Volume serial . .         ( May be generic )

  VRS name  . . . .
```

Figure 11-18 DFSMSrmm Add Vital Record Specification panel in z/OS Release 1.8 or higher

11.2.1 Data set VRS

A *data set VRS* defines the retention criteria by setting the total number of data set cycles to be retained, or the total number of days during which a data set needs to be retained. A data set *cycle* is a new version of the data set, so the number of data set cycles is equivalent to the number of data set versions.

You can combine or merge retention and movement policies, so you need to define more than one VRS for this data set. In this case, we call the *primary* VRS to the one that matches on the following values:

- Data set name
- Data set name mask
- Management class name or VRS management value

The *secondary* VRS matches on either management class or VRS management value.

A VRS *management value* is a single qualifier name you define in a VRS to tell DFSMSrmm how to manage and retain your tape data sets. The VRS management value can be up to eight alphanumeric characters, and must begin with an alphabetic character.

Figure 11-19 shows how DFSMSrmm performs vital record processing if you used the PARMLIB option VRSEL(NEW).

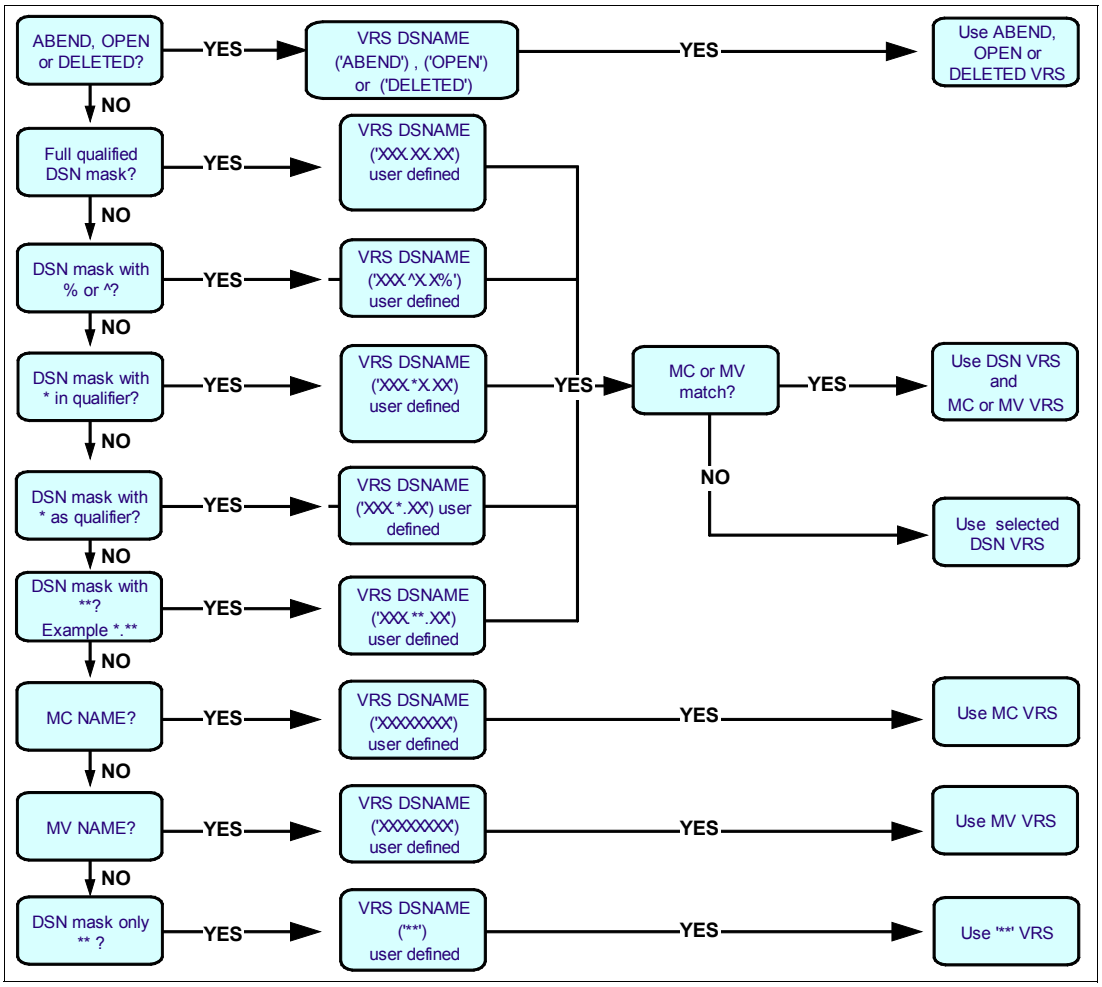


Figure 11-19 Selecting matching VRS for data sets (new)

VRSEL(NEW) provides the following new and improved functions:

- Mixed retention types: NEXTVRS to different type
- Combined retention: ANDVRS
- Release options: EXPIRYDATEIGNORE and SCRATCHIMMEDIATE
- Enable use of primary and secondary VRSs

- Management class and VRS management value are equivalents, and can be a secondary VRS policy

Matching data sets

Matching data sets have the following characteristics:

- Fully qualified data set names/job name

  The following VRS retains all copies of the data set DATA.SET.BACKUP for five days. NOGDG is the default.

  ```
  RMM ADDVRS DSNAME('DATA.SET.BACKUP') NOGDG DAYS COUNT(5)
  ```

- Data set name/job name masks

  You can use standard wildcards: %, *, **, and ¬.

  > **Note:** If a ¬ (not sign) is used in a data set name, the name is treated as a pseudo generation data group (GDG) entry.
  >
  > ** indicates to select all data sets. You can use this mask to define a VRS to define a retention criteria for data sets that are not covered by other VRSs.
  >
  > The data set name masks *.** and ** match to all data sets not covered by a more specific VRS. You can use these data set name masks to define a system-wide release option.

- Generation data groups (GDGs)

  You can use data set name or data set name masks with the GDG operand. In this case, the VRS only matches GDG data sets.

  The following VRS definition retains the last three versions of the GDG named DATA.SET.GDG:

  ```
  RMM ADDVRS DSNAME('DATA.SET.GDG') GDG CYCLE COUNT(3)
  ```

  But, it does not retain the non-GDG data set DATA.SET.GDG. You can also define a VRS definition using the not sign (¬) wildcard character to retain only GDGs as shown in this example:

  ```
  RMM ADDVRS DSNAME('DATA.SET.GDG.G¬¬¬¬V¬¬') CYCLE COUNT(3)
  ```

  The following VRS definition retains all GDGs starting with DATA.TEST:

  ```
  RMM ADDVRS DSNAME('DATA.TEST.**') GDG CYCLES COUNT(99999)
  ```

  The following VRS definition retains all GDGs and non-GDGs starting with DATA.TEST:

  ```
  RMM ADDVRS DSNAME(‘DATA.TEST.**’) NOGDG CYCLES COUNT(99999)
  ```

- Management class (**MC**)

  You can specify a management class name in a VRS. The MC is assigned by the storage management subsystem (SMS) automatic class selection (ACS) routines. For more information, see 11.7, “DFSMS ACS support” on page 414.

  The following example defines a VRS using a management class name to retain a data set with MC ‘DAYS003’. This MC is assigned to the data set by your installation through an ACS routine. The operand DAYS COUNT(3) indicates that the data set is to be retained for three days. Define a management class.

  ```
  RMM ADDVRS DSNAME('DAYS003') DAYS COUNT(3)
  ```

Note: There is no difference in the VRS definitions between a management class and VRS management value. Only the tools (SMS or EDGUX100) setting the value are different.

- VRS management value (**MV**)

  For data sets that do not belong to an MC, but need to be managed by special expiration dates, you can define a VRS management value. To assign MV to data sets, use the DFSMSrmm-supplied EDGUX100 installation exit.

  The following example defines a VRS to retain a data set with the special date 99000. The VRS MV 'CATALOG' is assigned by your installation in an installation exit, such as the EDGUX100 user exit. The WHILECATALOG operand indicates that the data set is to be retained while it is cataloged.

  ```
  RMM ADDVRS DSNAME('CATALOG') WHILECATALOG
  ```

- Status **ABEND**

  To control the retention of data sets closed as a result of an abnormal end in a task, you can define a VRS using a reserved data set name mask ABEND. For example, to retain the data set one day after the last reference, use the command that is shown in the following example:

  ```
  RMM ADDVRS DSNAME('ABEND') LASTREFERENCEDAYS COUNT(1)
  ```

  Note: You can define several ABEND VRSs to have different retention for data sets created using different JOBNAMEs:

  ```
  RMM ADDVRS DSNAME('ABEND') JOBNAME(*) DAYS COUNT(1)
  RMM ADDVRS DSNAME('ABEND') JOBNAME(TSM*) CYCLES COUNT(99999)
  RMM ADDVRS DSNAME('ABEND') JOBNAME(HSM*) CYCLES COUNT(99999)
  ```

  This retains all data sets created by using a job name, such as HSMA and HSMB, or TSM1, permanently. You must release them manually.

- Status **DELETED**

  Use this status to control the data sets created with a disposition of delete. DFSMSrmm provides support for managing deleted tape data sets. DFSMSrmm checks the normal disposition at CLOSE time. If this status is 'DELETE', a 'deleted' flag is recorded in the data set record. Subsequent use of a data set cannot change this flag. However, you can use the CHANGEDATASET command to reset the flag.

  You enable the management function by defining one or more DELETED VRSs. Deleted data sets that do not match to a DELETED VRS are matched to other VRSs normally:

  ```
  RMM AS DSNAME('DELETED') DAYS COUNT(0) -
         RELEASE(EXPIRYDATEIGNORE,SCRATCHIMMEDIATE)
  ```

  ```
  RMM AS DSNAME(mask) JOBNAME(DELETED) DAYS COUNT(0) -
         RELEASE(EXPIRYDATEIGNORE,SCRATCHIMMEDIATE):
  ```

Note: You can define several DELETED VRSs to have different retention for data sets created using different JOBNAMEs:

```
RMM ADDVRS DSNAME('DELETED') JOBNAME(*)  DAYS COUNT(1)
    RELEASE(EXPIRYDATEIGNORE,SCRATCHIMMEDIATE)
RMM ADDVRS DSNAME('**') JOBNAME(TSM*) CYCLES COUNT(99999)
RMM ADDVRS DSNAME('**') JOBNAME(HSM*) CYCLES COUNT(99999)
```

This retains all data sets created by using a job name, such as HSMA and HSMB, or TSM1 permanently. You must release them manually.

- Status **OPEN**

  To control the retention of data sets left open during a system failure, or that are in use during DFSMSrmm inventory management, use the reserved data set name mask OPEN. For example, to retain the data set two days after the last reference, use the command as shown in the following example:

  ```
  RMM ADDVRS DSNAME('OPEN') LASTREFERENCEDAYS COUNT(2)
  ```

  Note: You can define several OPEN VRSs to have different retention for data sets created using different JOBNAMEs:

  ```
  RMM ADDVRS DSNAME('ABEND') JOBNAME(*)  DAYS COUNT(1)
  RMM ADDVRS DSNAME('ABEND') JOBNAME(TSM*) CYCLES COUNT(99999)
  RMM ADDVRS DSNAME('ABEND') JOBNAME(HSM*) CYCLES COUNT(99999)
  ```

  This retains all data sets created by using a job name, such as HSMA and HSMB, or TSM1, permanently. You must release them manually.

Figure 11-20 on page 388 shows the order in which DFSMSrmm matches the data set name VRSs.

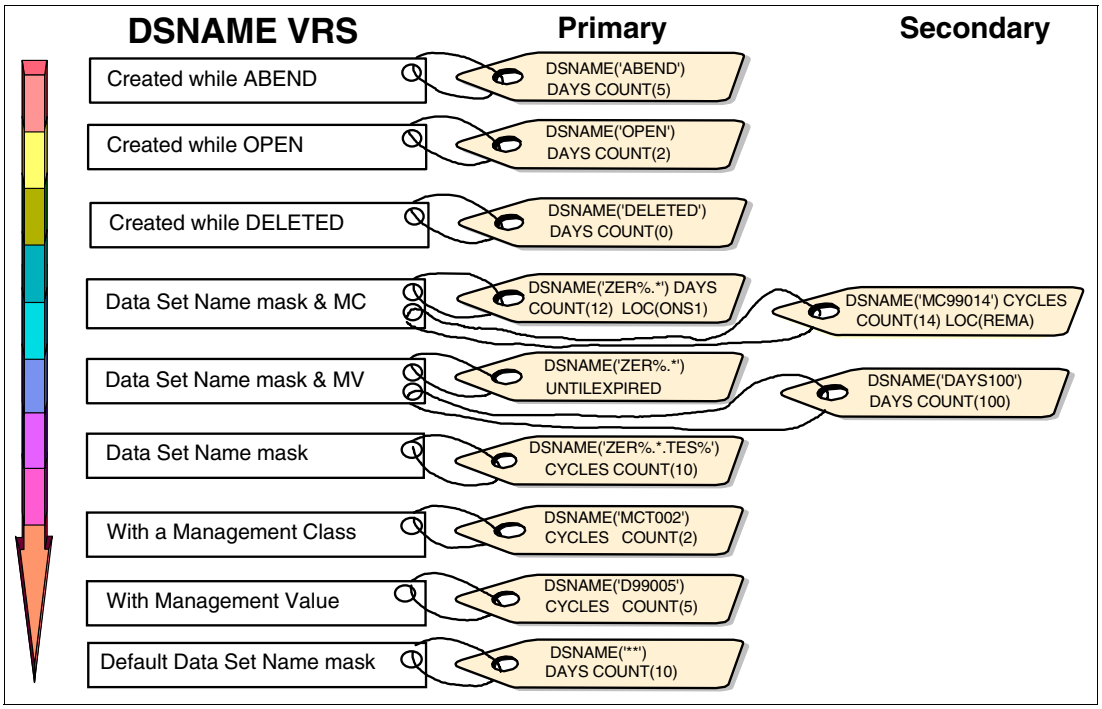


Figure 11-20 Data set name matching order

11.2.2 Volume VRS

A *volume VRS* defines the retention criteria by setting the total number of volumes to be retained, or the total number of days during which a volume needs to be retained.

Matching volumes

Volume VRSs are defined by using a specific or generic volume serial number. The matching rules for volumes are listed:

- Specific volume serial number

  Specify the volume serial number to retain only this volume. The following example shows how you can retain a specific volume for five days:

  ```
  RMM ADDVRS VOLUME(TST100) LOCATION(HOME) COUNT(5)
  ```

- Generic volumes

  You can specify generic volumes using an asterisk (*). The following example shows the RMM TSO ADDVRS subcommand that you can use to retain ten volumes matching the prefix TST:

  ```
  RMM ADDVRS VOLUME(TST*) LOCATION(REMOTE) COUNT(10)
  ```

11.2.3 Name VRS

Name VRSs can define additional movement criteria, and optionally define additional retention policy information. When a name VRS contains only movement information, it cannot exceed the retention criteria specified in the previous VRS, and it is called a *location* name VRS. We suggest that you do not use location name VRSs, but instead always specify a retention type and a COUNT value. You can define name VRSs for creating data set or volume VRS *chains*.

Specify the DFSMSrmm PARMLIB OPTION VRSEL(OLD) operand to define movement information only. Specify VRSEL(NEW) to define either retention or movement information, or both.

The following example shows the definition of a VRS chain where the second VRS is a name VRS with different retention criteria:

```
RMM ADDVRS DSN('DATA.SET.**') COUNT(3) CYCLES NEXTVRS(SECVRS)
RMM ADDVRS NAME(SECVRS) EXTRADAYS COUNT(10)
```

For more details about chaining VRSs, see 11.4, "Chaining VRSs" on page 393.

11.3 Specifying the VRS parameters

Table 11-1 shows the different parameters that you can specify in a VRS definition, depending on the type of VRS that you are defining.

Table 11-1 Specifying VRS

Types of VRS and retention	DATA SET	VOLUME	NAME
Match using	- Data set name - Management class - VRS management value - GDG - ABEND - OPEN	- Volume serial number - Generic VOLSER	
Retention types	- CYCLES - BYDAYSCYCLE - DAYS - LASTREFERENCE-DAYS - WHILECATALOG - UNTILEXPIRED	- Days - Volumes	- CYCLES - BYDAYSCYCLE - DAYS - LASTREFERENCE-DAYS - WHILECATALOG - UNTILEXPIRED - EXTRADAYS
Limits	- COUNT - DELETEDATE	- COUNT - DELETEDATE	- COUNT - DELETEDATE
Movement	- STORENUMBER - LOCATION - PRIORITY - DELAY	- STORENUMBER - LOCATION - PRIORITY - DELAY	- STORENUMBER - LOCATION
Release options	- SCRATCH-IMMEDIATE - EXPIRYDATE-IGNORE		
Chain using	- NEXTVRS - ANDVRS	- NEXTVRS	- NEXTVRS - ANDVRS
Chained from			- DSN VRS - VOLUME VRS - NAME VRS

11.3.1 DSNAME and NAME retention types

DFSMSrmm calculates the retention date for data sets based on a combination of the COUNT operand value and the retention type operands. The following retention type operands are available for data sets and names:

- **CYCLES**

  For GDG data sets, indicate the number of generations that DFSMSrmm must keep. To retain three versions of the GDG ITSO.GDG.DATA.SET, define the VRS as shown in the following example:

  ```
  RMM ADDVRS DSNAME('ITSO.GDG.DATA.SET') GDG CYCLES COUNT(3)
  ```

  For non-GDG data sets, each occurrence of a data set is a cycle. The following VRS retains three occurrences of the data set ITSO.NONGDG.DATA.SET:

  ```
  RMM ADDVRS DSNAME('ITSO.NONGDG.DATA.SET') CYCLES COUNT(3)
  ```

- **BYDAYSCYCLE**

  DFSMSrmm considers all the data sets created in a single day to be a cycle.

- **DAYS**

  This operand indicates a period of elapsed days from the creation date. The following command shows how you can retain data sets with the high-level qualifier YCJRES that are created in the last 10 days:

  ```
  RMM ADDVRS DSNAME('YCJRES.**') DAYS COUNT(10)
  ```

- **LASTREFERENCEDAYS**

  This operand indicates a period of elapsed days since the data set was last read or written to. You can define a VRS by using this command:

  ```
  RMM ADDVRS DSNAME('YCJRES.**') LASTREFERENCEDAYS COUNT(10)
  ```

  This command retains all data sets beginning by YCJRES that were last referenced during the last 10 days.

- **WHILECATALOG**

  You can use this operand to retain data sets while they are cataloged.

- **UNTILEXPIRED**

  A data set now optionally can be managed separately for retention. A VRS management value or a storage management subsystem (SMS) management class can identify retention criteria, and the data set name can be used to identify movement criteria. Up to two VRSs can be used to provide policy information for a data set.

  When new data is created, and no VRS management value or management class is assigned, because the jobs do not specify keyword dates and are not SMS managed, VRSEL processing finds only the data set VRS to match to, which specifies UNTILEXPIRED. The following example shows how you can define them.

  ```
  RMM ADDVRS DSNAME('**') UNTILEXPIRED
  ```

  DFSMSrmm checks the volume expiration date, and the UNTILEXPIRED option makes DFSMSrmm drop the data set from retention.

  The volume expiration date is also used if the following conditions are true and the PARMLIB option VRSEL(NEW) is specified:

  - When a data set matches a single VRS
  - When UNTILEXPIRED is used in the secondary VRS

The VRS enhancements introduced with APAR OW30969 enable “until expired” retention to be based on any type of retention. When a primary VRS and a secondary VRS match a data set, the use of UNTILEXPIRED in the primary uses the secondary VRS chain to determine overall retention. If there is no secondary VRS, or UNTILEXPIRED is used in a secondary VRS or a VRS management value VRS and PARMLIB option VRSEL(NEW) is used, only the volume expiration date is used. If VRSEL(OLD) is specified in PARMLIB, UNTILEXPIRED retention uses the volume expiration date or catalog status to determine how long to retain a data set.

Any policy used for retention or movement, or both retention and movement, can be a chain of multiple VRSs, where each VRS in a chain can use a different type of retention. Both the primary and secondary VRSs can be VRS chains, offering enough flexibility to create quite complex retention and movement policies.

A secondary VRS can be only a VRS management class or a VRS management value.

11.3.2 Special NAME retention type

If you specified to use a NEXTVRS in chain, you can use the EXTRADAYS operand on it. On a NEXTVRS, you can specify the number of days since a name VRS started to retain the data set. The number of extra days is specified with the COUNT operand.

EXTRADAYS is the number of days since the NAME VRS started to retain the data set. The number of days depends on when the previous VRS stopped retaining the data set, and the time when vital record processing is run. This option can only be used on a NEXTVRS and not a primary VRS.

11.3.3 Volume retention types

DFSMSrmm calculates the retention date for volumes based on a combination of the COUNT operand value and the retention type operands. Volumes can have these retention type operands:

- By **DAYS**

  By the number of elapsed days since creation, which indicates that a volume is retained until the specified number of days since creation has elapsed.

  To retain volume V20001 for 31 days in a storage location, type the following command:

  ```
  RMM ADDVRS VOLUME(V20001) DAYS COUNT(31) LOCATION(DISTANT)
  ```

- By the number of volumes (**COUNT**)

  Indicates how many volumes to retain. For example, to retain the last 50 volumes matching the generic volume serial number, TST, in a storage location, type the following command:

  ```
  RMM ADDVRS VOLUME(TST*) COUNT(50) LOCATION(DISTANT)
  ```

11.3.4 Retention limits

You can define the limits for a volume retention or for a VRS to be considered as valid. You can use the following parameters to specify the limits:

- **COUNT**

  Specifies a retention amount, based on the retention type chosen: number of days or cycles for data sets VRS, and number of volumes for volume VRS. The COUNT default is 99999.

You can specify COUNT with DAYS to indicate the number of days that DFSMSrmm retains your data set, as shown in the following example:

```
RMM ADDVRS DSNAME('YCJRES.**') DAYS COUNT(10)
```

Or, you can specify COUNT with CYCLES to indicate the number of cycles of the data set that DFSMSrmm retains, as shown in the following example:

```
RMM ADDVRS DSNAME('YCJRES.BACKUP.CYCLE') CYCLES COUNT(8)
```

Note: A value of 99999 indicates that DFSMSrmm retains all cycles of a data set, or all volumes specified.

- **DELETEDATE**

  Specifies the date when DFSMSrmm deletes the VRS. The default value is 1999/365, which indicates that DFSMSrmm never deletes the VRS. When that date is reached, all matching data sets and volumes are eligible for matching to other, less specific VRSs, and if no VRS is retained, the volumes are eligible for release processing.

  The following example shows defining how a VRS retains the data set until the VRS deletion date (2005/170) is reached, or the data set is uncataloged:

```
RMM ADDVRS DSNAME('YCJRES1.DELETE.DATE')     -
          DELETEDATE(2005/170) WHILECATALOG
```

Note: If you have a DELETEDATE specified, and the date is reached, all data sets matching this VRS are never passed to another VRS.

11.3.5 Movement policies

You can define policies to provide retention information for data sets and volumes that must be moved through multiple locations before they expire. You define such policies by creating a VRS chain and using the following parameters:

STORENUMBER For data set VRS, this parameter specifies how many days to retain a data set, and how many data set cycles or versions to retain in the location of the volume. For volume VRS, this parameter specifies how many volumes to retain, or how many days to retain a volume in the location specified in the parameter LOCATION. The number specified must be less than or equal to COUNT.

LOCATION Specifies a location where the volume will be retained. Use HOME if you want the volume to be returned to its home location. Use CURRENT to avoid moving volumes.

PRIORITY When multiple data sets on the same volume are retained by VRSs, and each VRS contains a different destination, DFSMSrmm decides where to move the volume according to this priority number. The default value 0 means that the priority specified in the LOCDEF parameter on PARMLIB needs to be used.

DELAY Specifies the number of days that a data set or volume is retained in its current location before it is sent to a storage location.

11.3.6 Release options

You can specify special release options to be processed after a data set is dropped from VRS, and the volume is released. Release options take effect only for data sets after they have been retained by a VRS.

SCRATCHIMMEDIATE
If you specify this option, DFSMSrmm returns the volume to scratch status in a single run of inventory management (EXPROC processing) when the only pending release action for the volume is to return to scratch, and you have specified VRSEL(NEW).

EXPIRYDATEIGNORE
If you specify this option, DFSMSrmm ignores the volume expiration date when VRSEL(NEW) is used and no other VRS matches the data set.

11.4 Chaining VRSs

A *VRS chain* consists of a volume or data set VRS, and all of the name VRSs chained from it. For chaining two or more VRS, you must use one of the following operands:

NEXTVRS Specify NEXTVRS(*name*) to point to another VRS, and create a chain of VRSs. With NEXTVRS, DFSMSrmm processes each VRS in the chain separately. You can only chain volume VRSs with the NEXTVRS operand.

ANDVRS Specify ANDVRS to create a chain of VRSs where all the retention conditions must be true for the data sets to be retained. DFSMSrmm uses the STORENUMBER and LOCATION from the first VRS in the VRS chain.

You can combine VRSs using NEXTVRS and ANDVRS operands; these combinations are called VRS chain and VRS subchain:

- A *VRS chain* is a volume or data set VRS and all name VRSs chained from it.
- A *VRS subchain* is a data set VRS, name VRS with retention information, or ANDVRS group, and all the VRSs chained from it, without including the NEXTVRS in the chain that contains retention information.

Both a VRS chain and subchain can include one or more VRSs. When a data set name VRS is chained only to name VRSs that have no retention information, there is no difference between a VRS chain and a VRS subchain. The following examples explain the differences between chain and subchain VRSs.

Figure 11-21 shows a VRS chain with three subchains. They are a data set VRS with NEXTVRS, and two name VRSs with retention information.

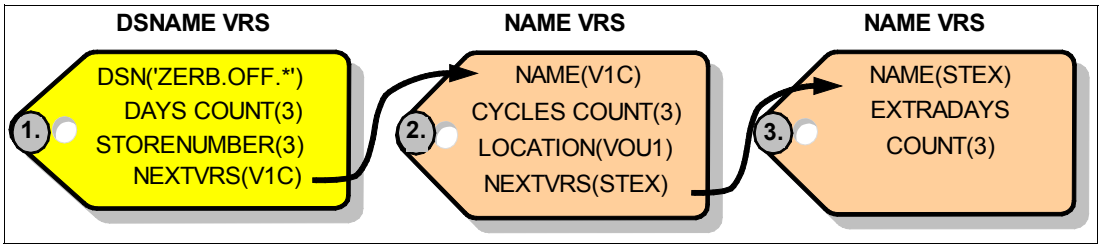


Figure 11-21 VRS chain with three subchains

Figure 11-22 shows a VRS chain with two subchains. It is a data set VRS with an ANDVRS that form one subchain, and a name VRS joined with a NEXTVRS command with retention information.

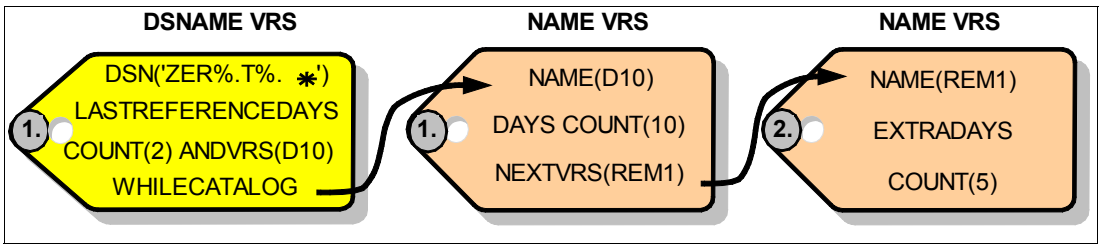


Figure 11-22 VRS chain with two subchains

Figure 11-23 shows a VRS chain with a single subchain; the VRSs are joined by ANDVRS commands.

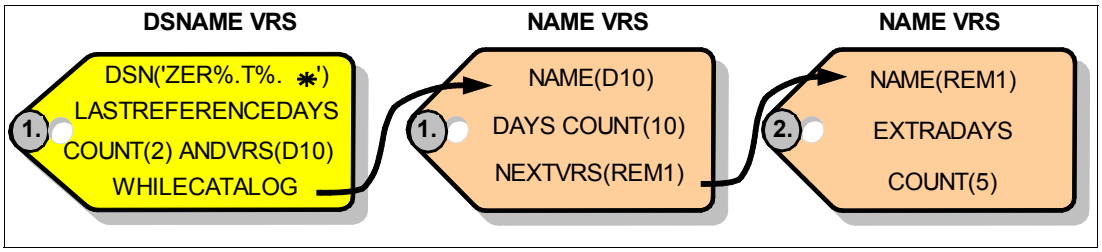


Figure 11-23 VRS chain using NEXTVRS

Figure 11-24 shows a VRS chain with two subchains. The data set VRS and the first name VRS form a subchain, because the name VRS does not have retention information. The second name VRS is a different VRS subchain.

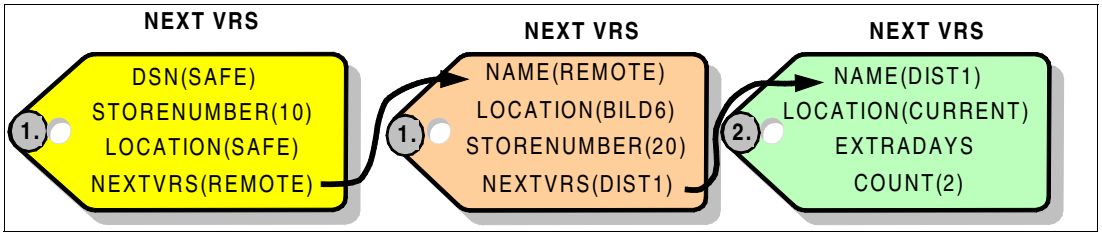


Figure 11-24 VRS chain with two subchains

11.4.1 VRS chain and subchain

The VRS is key to how DFSMSrmm manages your tape data sets and volumes; retention and movement are driven by VRSs. There are many ways to build VRS chains and subchains. Even if you are able to meet all your retention and movement needs, your approach needs to be to keep the VRS chain and subchain as simple as possible.

You need to know the following terms before we describe building VRS chains and subchains:

Data set group All the data sets with a name that matches a VRS are treated as a single vital retention group. You can use a GDG base name when defining a VRS to retain volumes. You must not supply the generation data set group suffix. You must specify `CYCLES` if you want DFSMSrmm to manage the data sets as a data set group. If you are using GDG version numbering, DFSMSrmm only keeps the latest version of each generation.

VRS chain A volume or data set VRS and all of the name VRSs chained from it.

VRS subchain A data set name VRS, NAME VRS with retention information, or AND VRS group and all the VRSs chained from it, up to but not including the next VRS in the chain that contains retention information. Both a VRS chain and subchain can consist of one or more VRSs. When a data set name VRS is chained only to NAME VRSs that have no retention information, there is no difference between a VRS chain and a VRS subchain.

Note: A data set VRS can only be used as the first definition in a chain. A data set VRS cannot be used in the NEXTVRS operand.

11.5 VRSEL GDG option

New DFSMSrmm parmlib options provide flexibility in how tape generation data sets are managed for cyclic retention.

Use the GDG option to specify how generation data groups are handled for cycle retention by VRSEL processing. Cycle retention includes both the CYCLES and the BYDAYSCYCLE retention types. The correct sequence for determining the retention can be either by using the generation number or the creation order. You can also specify how duplicate generations are handled. You have the flexibility to include or exclude duplicates from the cycles count as required by your application processing.

The correct syntax for the new GDG OPTION command is shown in Figure 11-25.

```
                        |--GDG(CYCLEBY(GENERATION)DUPLICATE(BUMP))--|
>>------ OPTION ------------------------------------------------------------<<
                        |                                           |
                        |--GDG(----gdg_options-)--------------------|

   gdg_options
>------------------------------------------------------------------------------>
   |                          |      |                          |
   |          |-GENERATION-|  |      |              |-BUMP--|   |
   |--CYCLEBY(--------------)--|      |----DUPLICATE(----------)--|
              |-CRDATE-----|                         |-DROP--|
                                                     |-KEEP--|
                                                     |-COUNT-|
```

Figure 11-25 GDG OPTION command syntax

The following definitions apply to the GDG OPTION command:

GDG

Use the GDG option to specify how generation data groups are handled for cycle retention by VRSEL processing. Cycle retention includes both the CYCLES and the BYDAYSCYCLE retention types. The correct sequence for determining the retention can be either by using the generation number or the creation order. You can also specify how duplicate generations (generation data sets) are handled and you have the flexibility to include or exclude duplicate generations from the cycles count as required by your application processing.

The GDG option has two operands, CYCLEBY and DUPLICATE.

Use the CYCLEBY operand to specify whether retention is based on generation number or creation date:

CYCLEBY(GENERATION)

Specifies that retention is based on the generation number. DFSMSrmm determines the generation number by applying a similar algorithm to that used by Catalog processing. Both the creation order and the generation number from the data set name are considered, allowing wraps in generation number to be correctly handled.

CYCLEBY(GENERATION) is the default.

CYCLEBY(CRDATE)

Specifies that only the creation sequence is to be used to determine the retention.

Use the DUPLICATE operand to specify how VRSEL processing handles duplicate generations. You can specify one of the following options:

- Count duplicate generations
- Keep duplicate generations, but do not count them
- Bump duplicate generations from the current subchain
- Drop duplicate generations from VRS retention

Duplicate generations are determined within a single VRS subchain and only if the generation and the generation it duplicates are not already dropped for another reason. For GENERATION-based cycles, the duplicate generations are determined in generation (data set name), then creation order. For creation date-based cycles, the duplicate generations are determined only in creation order so that they can be detected only when they are created consecutively.

Duplicates are processed within the context of the matching VRS chain or subchain, depending on the DUPLICATE option. DROP is within the context of the VRS chain and all others are within the context of the sub chain. A duplicate generation is considered a duplicate for cycles retention only if the generation it duplicates is retained by the current VRS sub chain.

DUPLICATE(BUMP)

Specifies that a duplicate generation is to be bumped by the current VRS subchain and considered for retention by a subsequent VRS subchain. DUPLICATE(BUMP) is the default.

DUPLICATE(COUNT) Specifies that a duplicate generation is to be treated like a non-duplicate generation regarding CYCLE and BYDAYSCYCLE processing.

DUPLICATE(DROP) Specifies that a duplicate generation is to be dropped from VRS retention without further consideration.

DUPLICATE(KEEP) Specifies that a duplicate generation is considered as either the same CYCLE or the same BYDAYSCYCLE depending on the VRS retention type and regardless of the duplicate generation creation date.

Before we describe the handling options that can be specified, we describe the circumstances under which VRSEL applies the GDG(DUPLICATE()) operand. The following numbers correspond to Figure 11-26:

1. Same generation number as previous.

   Consider GDG(CYCLEBY(CRDATE)). A higher generation might have already been created before the same generation has been created again (like the example shown in Figure 11-26).

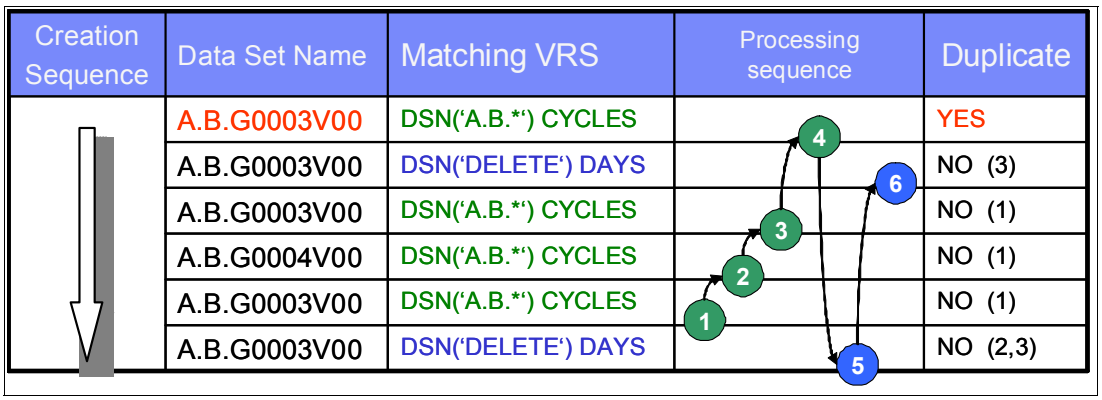
2. Same matching VRS as previous (same subchain).

   Reasons why data sets with the same name and generation number match to a different VRS:

   – DELETE VRS, OPEN VRS, or ABEND VRS
   – VRS with Management Class
   – VRS with JOBNAME()

   DUPLICATE processing only applies within a single VRS subchain.

3. VRS retention type is CYCLES or BYDAYSCYCLE.
4. Met other retention criteria (for example, UNTILEXPIRED).



Creation Sequence	Data Set Name	Matching VRS	Processing sequence	Duplicate
	A.B.G0003V00	DSN('A.B.*') CYCLES	4	YES
	A.B.G0003V00	DSN('DELETE') DAYS	6	NO (3)
	A.B.G0003V00	DSN('A.B.*') CYCLES	3	NO (1)
	A.B.G0004V00	DSN('A.B.*') CYCLES	2	NO (1)
	A.B.G0003V00	DSN('A.B.*') CYCLES	1	NO (1)
	A.B.G0003V00	DSN('DELETE') DAYS	5	NO (2,3)

Figure 11-26 Process data set as duplicate generation

The tables in Figure 11-27 on page 398 shows the different results of the DUPLICATE operands BUMP, DROP, KEEP, and COUNT. Assume that three generation data sets exist in a cycle group. The "red" generation 3 is processed as a "duplicate". The matching VRS has two subchains. The first one keeps two cycles. The second one has a DAYS retention.

BUMP The duplicate does not get retained by the CYCLES subchain, but the next subchain retains it (assuming that the retention criteria are met).

DROP The duplicate is dropped from the whole VRS chain.

KEEP The duplicate gets retained by the CYCLES subchain as the same cycle as its original, so that three data sets are kept with COUNT(2).

COUNT The duplicate is also retained by the CYCLES subchain, however, as its own cycle, so that generation 2 cannot be kept in this subchain anymore.

DUPLICATE processing works similarly to CYCLES and BYDAYSCYCLE retention. For BYDAYSCYCLE, if the duplicate is created on the same day as its original, KEEP and COUNT make no difference because, for BYDAYSCYCLE, data sets that are created on the same day are treated as one cycle.

VRS Subchains	GDG DUPLICATE(...)			
	BUMP	DROP	KEEP	COUNT
CYCLES COUNT(2)	A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0003V00 A.B.G0002V00	A.B.G0003V00 A.B.G0003V00
DAYS COUNT(100)	A.B.G0003V00			A.B.G0002V00

Figure 11-27 Different results of the DUPLICATE operands BUMP, DROP, KEEP, and COUNT

Use the RMM TSO LISTCONTROL OPTION to see the current parmlib option setting. Figure 11-28 shows the correct use of the command.

```
RMM LISTCONTROL OPTION
```

Figure 11-28 LISTCONTROL OPTION

Example 11-1 shows the result of the LISTCONTROL command.

Example 11-1 Result of LISTCONTROL

```
System options:
System options:
PARMLIB Suffix  = 02
Operating mode  = P    Retention period: Default = 0       Maximum = NOLIMIT
                                         Catalog = 6     hours
                       Retention method: Method  = VRSEL
Control data set name      = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold          = 75%               Journal transaction = NO
Catalog SYSID              = Notset
Scratch procedure name     = EDGXPROC
Backup procedure name      = EDGCDSBK
IPL date check  = N        Date format    = J       RACF support   = N
SMF audit       = 0        SMF security   = 42      CDS id         = SC70
MAXHOLD value   = 100      Lines per page = 54      System ID      = SC70
BLP             = RMM      TVEXT purge    = RELEASE Notify         = N
                                  days    = 0
Uncatalog       = Y        VRS job name   = 2       Message case   = M
MASTER overwrite= USER     Accounting     = J       VRS selection  = NEW
VRS change      = INFO     GDG duplicate  = BUMP    GDG cycle by = GENERATION
VRSMIN action   = INFO     VRSMIN count   = 1
VRSDROP action  = INFO     VRSDROP count  = 0              percent = 10
```

```
VRSRETAIN action= INFO     VRSRETAIN count= 0              percent = 80
EXPDTDROP action= INFO     EXPDTDROP count= 0              percent = 10
Disp DD name    = DISPDD   Disp msg ID    = EDG4054I
Retain by       = SET      Move by        = SET     CMDAUTH Owner  = NO
PREACS          = NO       SMSACS         = NO      CMDAUTH Dsn    = YES
Reuse bin       = CONFIRMMOVE                       Media name     = 3480
                           Local tasks   = 10   PDA: ON
 Block count    = 0        Block size     = 0       Log            = ON
SMSTAPE:
 Update scratch = YES      Update command = YES     Update exits   = YES
 Purge          = ASIS
Client/Server:
 Subsystem type = STANDARD Port           = 0
 Server                    Server tasks   = 0
  host name  =
  IP address =
```

The activity log now shows two additional fields as shown in Figure 11-29. These fields show the active GDG definitions at this run.

```
----+----1----+...---7----+----8----+----9----+----0----+----1----+----2-
H   2009/223  1...SC70    NSS         0 10I         0 80I         0 10IGB
```

Figure 11-29 Activity header record

The following information applies to Figure 11-29:

GDGCYCLEBY On column 120:

```
ACTRC_HDR_GDGCYCLEBY      DS  C     GDG(CYCLEBY()
ACTRC_HDR_GDGC_GENERATION EQU C'G'
ACTRC_HDR_GDGC_CRDATE     EQU C'C'
```

GDGDUPLICATE On column 121:

```
ACTRC_HDR_GDGDUPLICATE    DS  C     GDG(DUPLICATE()
ACTRC_HDR_GDGD_BUMP       EQU C'B'
ACTRC_HDR_GDGD_DROP       EQU C'D'
ACTRC_HDR_GDGD_KEEP       EQUC'K'
ACTRC_HDR_GDGD_COUNT      EQU C'C'
```

11.5.1 Generation data set and DSN VRS creation

To test and show you how this new functionality works, we created 12 data sets on tape and use these data sets always in our test cases. See Table 11-2 for the details of the created data sets. We created each generation data set twice with different creation days. You can also see that the generation data sets G0003V00 are created after the generation data sets G0004V00.

Table 11-2 Created data sets to test the new GDG parmlib option

DSNAME	VOLSER	CRDATE
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00	NS0001	2009/200
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00	NS0011	2009/201
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00	NS0002	2009/207
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00	NS0012	2009/208
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00	NS0003	2009/221

DSNAME	VOLSER	CRDATE
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00	NS0013	2009/222
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00	NS0004	2009/214
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00	NS0014	2009/215
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00	NS0005	2009/228
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00	NS0015	2009/229
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00	NS0006	2009/235
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00	NS0016	2009/236

Next, we have a DSNAME VRS with the attributes to match the data sets shown in Table 11-2 on page 399. See Figure 11-30.

```
                        DFSMSrmm Display Data Set VRS
Command ===>

Data set mask . : 'RMM.TEST.NEW.GDG.PARMLIB.OPTION.**'          GDG . : NO
     Job name mask . :

Count . . . : 5                     Retention type  . . . . . : CYCLES
                                    While cataloged . . . . . : NO
Delay . . . : 0   Days              Until expired . . . . . . : NO

Location  . . . . . : HOME
Number in location  : 5
Priority  . . . . . : 0
                                    Release options:
Next VRS in chain . : GDGOPT        Expiry date ignore  . . . : NO
    Chain using . . :               Scratch immediate . . . . : NO

Owner . . . . . : SCHLUM
Description . . : RETAIN AND MOVE DATA SETS
Last reference  :                       ( YYYY/DDD  HH:MM:SS )
Delete date . . : 1999/365    ( YYYY/DDD )

Last Change information:
 Date . . . . . . : 2010/289       Time  . : 20:45:17     System  : SC70
 User change date : 2010/289       Time  . : 20:45:17     User ID : MHLRES7
```

Figure 11-30 DSNAME VRS to test new GDG parmlib option

To show the differences better, we also added a management value/management class VRS. See Figure 11-31 on page 401 for the details of this additional DSNAME VRS.

```
                          DFSMSrmm Display Name VRS
Command ===>

Name  . . . . . : GDGOPT

Count . . . : 6                  Retention type  . . . . . : CYCLES
                                 While cataloged . . . . . : NO
                                 Until expired . . . . . . : NO
Location  . . . . . : HOME
Number in location  : 6

Next VRS in chain . :
    Chain using . . :

Owner . . . . . : SCHLUM
Description . . : TEST NEW GDG PARMLIB OPTION
Last reference  :                        ( YYYY/DDD HH:MM:SS )
Delete date . . : 1999/365    ( YYYY/DDD )

Last Change information:
 Date . . . . . . : 2011/335       Time  . : 12:49:20     System  : SC70
 User change date : 2011/335       Time  . : 12:49:20     User ID : MHLRES7
```

Figure 11-31 Management value/management class VRS definition

11.5.2 CYCLEBY GENERATION and DUPLICATE BUMP

Figure 11-32 shows the normal GDG processing as it was used before. It matched the GDG parmlib option GDG(CYCLEBY(GENERATION) DUPLICATE(BUMP)). It is the default if you have not specified the GDG option in your EDGRMMnn parmlib. You can also specify CYCLEBY(CRDATE) to fulfill the old processing if you created the data sets in sequence.

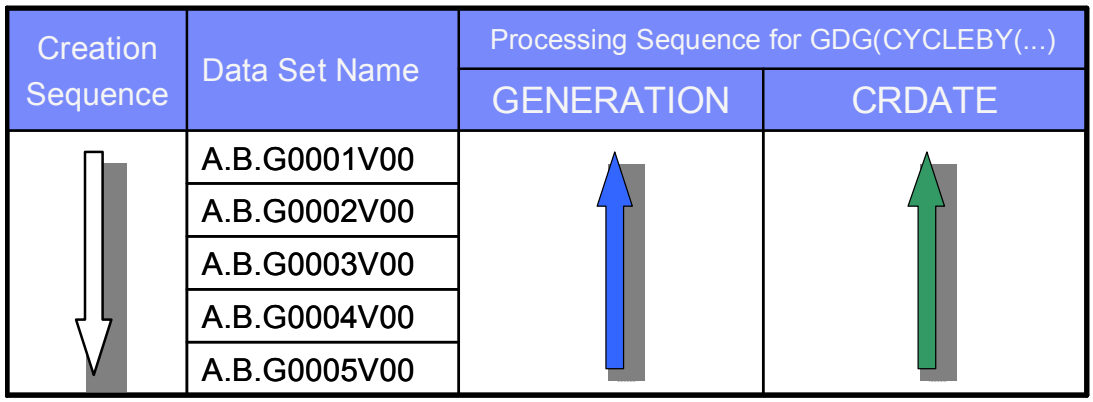


Figure 11-32 Generations created consecutively

In our first test case, we created 12 files. We specified a DSNAME VRS with the option CYCLES and a count of 5 to show you how it works. In Figure 11-33 on page 402, you can see the detail DSNAME VRS options.

We created each generation data set twice to show you that only the most current generation data set is retained and the older one is not. Table 11-2 on page 399 shows all the data sets

that we created. Figure 11-33 shows how a generation data set is retained and how it is not. Only one circumstance of the same generation data set is retained.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                    PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                         TIME 14:07:04 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                               MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                         11     0
```

Figure 11-33 Result of test case one

In Table 11-3, you can see the drop reason and why a data set is retained.

Table 11-3 Details of test case one

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	**Next**
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Duplicate GDS**
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G0002V00	NS0002	2009/207	Next	10	**Duplicate GDS**	CYCL/00006 5
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G0003V00	NS0003	2009/221	Next	8	**Duplicate GDS**	CYCL/00006 4
RMM.....G0003V00	NS0013	2009/222	Prim	7	CYCL/00005 4	
RMM.....G0004V00	NS0004	2009/214	Next	6	**Duplicate GDS**	CYCL/00006 3
RMM.....G0004V00	NS0014	2009/215	Prim	5	CYCL/00005 3	
RMM.....G0005V00	NS0005	2009/228	Next	4	**Duplicate GDS**	CYCL/00006 2
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	Next	2	**Duplicate GDS**	CYCL/00006 1
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.3 CYCLEBY CRDATE and DUPLICATE BUMP

If, however, generation data sets are created out of sequence, you can use the GDG(CYCLEBY(CRDATE)) operand of the parmlib OPTION command to specify whether VRSEL processes the data sets in generation sequence from the highest number to the lowest number.

Or, you can specify GDG(CYCLEBY(CRDATE)) to process the data sets in creation date sequence starting from the newest (youngest). Figure 11-34 shows the sequence in which the data sets are created.

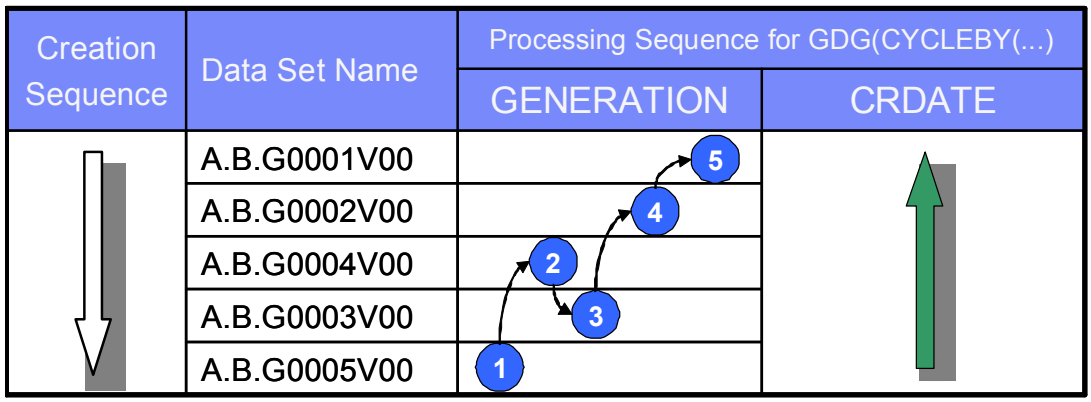


Figure 11-34 Generations out of sequence

In our second test case, we used the same 12 files as before, but we have changed the GDG option from CYCLESBY(GENERATION) to CYCLESBY(CRDATE). Figure 11-35 shows the result of our test.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                    PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                        TIME 14:09:45 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME          2009/223
          GDGOPT                              MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME          2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                          11     0
```

Figure 11-35 Result of test case two

In Table 11-4, you can see the drop reason and why a data set is retained. Generation data set G0003V00 is retained before generation data set G0004V00 depending on the creation date.

Table 11-4 Details of test case two

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	**Next**
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Duplicate GDS**
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G0002V00	NS0002	2009/207	Next	10	**Duplicate GDS**	CYCL/00006 5

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G0003V00	NS0003	2009/221	Next	**6**	**Duplicate GDS**	CYCL/00006 3
RMM.....G0003V00	NS0013	2009/222	Prim	**5**	CYCL/00005 3	
RMM.....G0004V00	NS0004	2009/214	Next	**8**	**Duplicate GDS**	CYCL/00006 4
RMM.....G0004V00	NS0014	2009/215	Prim	**7**	CYCL/00005 4	
RMM.....G0005V00	NS0005	2009/228	Next	4	**Duplicate GDS**	CYCL/00006 2
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	Next	2	**Duplicate GDS**	CYCL/00006 1
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.4 CYCLEBY GENERATION and DUPLICATE BUMP with generation wrap

Now, we have simulated the generation wrap-up. Therefore, we have reached the maximum count of 9999 of a generation data set and the next generation is created with a version of 1.

In our third test case, we created 12 files (similar to the files we created before), as shown in Table 11-5. However, we started with the generation data set G9997V00 so that we get a generation wrap. In Figure 11-30 on page 400, you can see the detailed DSNAME VRS options. In Figure 11-31 on page 401, you can see the NAME VRS details.

Table 11-5 Created a data set with a generation wrap to test new GDG parmlib option

DSNAME	VOLSER	CRDATE
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9997V00	NS0001	2009/200
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9997V00	NS0011	2009/201
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00	NS0002	2009/207
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00	NS0012	2009/208
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00	NS0003	2009/221
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00	NS0013	2009/222
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00	NS0004	2009/214
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00	NS0014	2009/215
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00	NS0005	2009/228
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00	NS0015	2009/229
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00	NS0006	2009/235
RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00	NS0016	2009/236

Figure 11-36 on page 405 illustrates a generation wrap that means that you have reached the maximum of 9999 for generation data set and the data set create gets the number 0001.

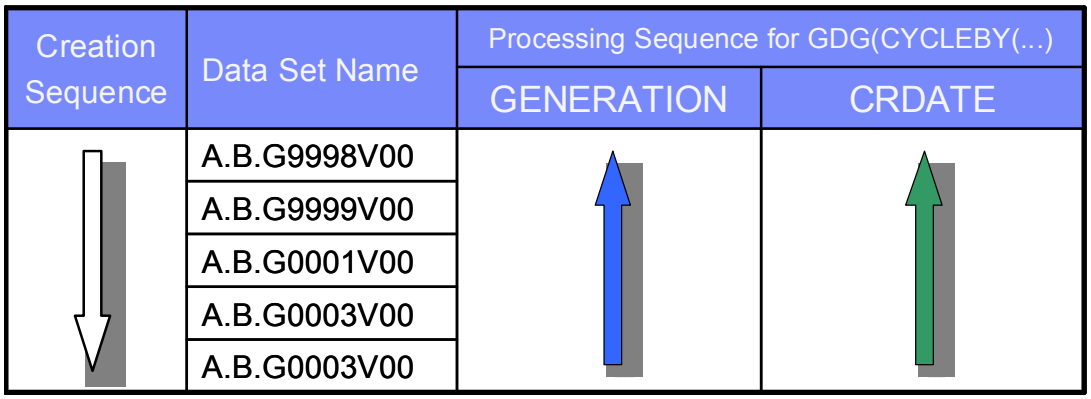


Figure 11-36 Generation wrap

Similar to the first test case, only the most recent generation data set is retained by the primary VRS definition and the older one is not retained. Figure 11-37 shows all the data sets retained in this test case.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                  PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                        TIME 14:08:14 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                              MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                         FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9997V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                        11     0
```

Figure 11-37 Result of test case three

In Table 11-6, you can see the drop reason and why a data set is retained.

Table 11-6 Details of test case three

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	**Next**
RMM.....G9997V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Duplicate GDS**
RMM.....G9997V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G9998V00	NS0002	2009/207	Next	10	**Duplicate GDS**	CYCL/00006 5
RMM.....G9998V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G9999V00	NS0003	2009/221	Next	8	**Duplicate GDS**	CYCL/00006 4
RMM.....G9999V00	NS0013	2009/222	Prim	7	CYCL/00005 4	
RMM.....G0001V00	NS0004	2009/214	Next	6	**Duplicate GDS**	CYCL/00006 3

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0014	2009/215	Prim	5	CYCL/00005 3	
RMM.....G0001V00	NS0005	2009/228	Next	4	**Duplicate GDS**	CYCL/00006 2
RMM.....G0001V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0001V00	NS0006	2009/235	Next	2	**Duplicate GDS**	CYCL/00006 1
RMM.....G0001V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.5 CYCLEBY CRDATE and DUPLICATE BUMP with generation wrap

In our fourth test case, we used the same 12 files as before, but we changed the GDG option from CYCLESBY(GENERATION) to CYCLESBY(CRDATE). As shown in Figure 11-38, the generation data set G9999V00 is retained before generation data set G0001V00 based on the creation date.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                        TIME 14:08:14 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                              MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9999V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9998V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G9997V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                         11     0
```

Figure 11-38 Result of test case four

In Table 11-7, you can see the drop reason and why a data set is retained. Generation data set G9999V00 is retained before generation data set G0001V00 based on the creation date.

Table 11-7 Details of test case four

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G9997V00	NS0001	2009/200	No	12	**Cycles exceeded**	**duplicate GDS**
RMM.....G9997V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G9998V00	NS0002	2009/207	Next	10	**Duplicate GDS**	CYCL/00006 5
RMM.....G9998V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G9999V00	NS0003	2009/221	Next	6	**Duplicate GDS**	CYCL/00006 3

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G9999V00	NS0013	2009/222	Prim	5	CYCL/00005 3	
RMM.....G0001V00	NS0004	2009/214	Next	8	**Duplicate GDS**	CYCL/00006 4
RMM.....G0001V00	NS0014	2009/215	Prim	7	CYCL/00005 4	
RMM.....G0001V00	NS0005	2009/228	Next	4	**Duplicate GDS**	CYCL/00006 2
RMM.....G0001V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0001V00	NS0006	2009/235	Next	2	**Duplicate GDS**	CYCL/00006 1
RMM.....G0001V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.6 CYCLEBY GENERATION and DUPLICATE DROP

In our fifth test, we changed the GDG parmlib option to GDG(CYCLEBY(GENERATION) DUPLICATE(DROP)). Figure 11-39 shows that only the newer version of each generation data set is retained. Because of the DROP option that we specified, the older versions are dropped by cycles exceeded.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                        TIME 14:09:45 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                               MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                                    6      0
```

Figure 11-39 Result of test case five

Table 11-8 shows the drop reason and why a data set is retained.

Table 11-8 Details of test case five

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Duplicate GDS**
RMM.....G0001V00	NS0011	2009/201	Next	11	Cycles exceeded	CYCL/00006 1
RMM.....G0002V00	NS0002	2009/207	No	10	**Duplicate GDS**	
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G0003V00	NS0003	2009/221	No	8	**Duplicate GDS**	
RMM.....G0003V00	NS0013	2009/222	Prim	7	CYCL/00005 4	

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0004V00	NS0004	2009/214	Not	6	**Duplicate GDS**	
RMM.....G0004V00	NS0014	2009/215	Prim	5	CYCL/00005 3	
RMM.....G0005V00	NS0005	2009/228	No	4	**Duplicate GDS**	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	No	2	**Duplicate GDS**	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.7 CYCLEBY CRDATE and DUPLICATE DROP

In our sixth test, we changed the GDG parmlib option to GDG(CYCLEBY(GENERATION) DUPLICATE(DROP)). Figure 11-40 shows that only the newer version of each generation data set is retained on the DROP specification. Because of the DROP option that we specified, the older versions are dropped by cycles exceeded, but generation data set 4 will be retained before generation data set 3.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                        TIME 14:09:45 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK               OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF
_____________________________________________________________________________________________________________________________
          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**   SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                               MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME
_____________________________________________________________________________________________________________________________
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                                   6      0
```

Figure 11-40 Result of test case six

Table 11-9 shows the drop reason and why a data set is retained. Generation data set G0003V00 is retained before generation data set G0004V00 because of the creation date.

Table 11-9 Details of test case six

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Duplicate GDS**
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 1
RMM.....G0002V00	NS0002	2009/207	No	10	**Duplicate GDS**	
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G0003V00	NS0003	2009/221	No	**6**	**Duplicate GDS**	

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0003V00	NS0013	2009/222	Prim	**5**	CYCL/00005 4	
RMM.....G0004V00	NS0004	2009/214	No	**8**	**Duplicate GDS**	
RMM.....G0004V00	NS0014	2009/215	Prim	**7**	CYCL/00005 3	
RMM.....G0005V00	NS0005	2009/228	No	4	**Duplicate GDS**	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	No	2	**Duplicate GDS**	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.8 CYCLEBY GENERATION and DUPLICATE KEEP

In our seventh test, we changed the GDG parmlib option to GDG(CYCLEBY(GENERATION) DUPLICATE(KEEP)). Figure 11-41 shows that both versions of the current five generation data sets are retained by the primary VRS definition and that both versions of the oldest generation data sets are retained by the next VRS definition, GDGOPT.

```
REMOVABLE MEDIA MANAGER                  VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007        ----- ------- --------- ------                        TIME 14:09:18 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK                OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**     SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                                 MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0001    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                         12     0
```

Figure 11-41 Result of test case seven

Table 11-10 shows that all 12 data sets are retained.

Table 11-10 Details of test case seven

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0001	2009/200	Next	12	**Cycles exceeded**	CYCL/00006 1
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 1
RMM.....G0002V00	NS0002	2009/207	Next	10	CYCL/00005 5	
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0003V00	NS0003	2009/221	Prim	8	CYCL/00005 4	
RMM.....G0003V00	NS0013	2009/222	Prim	7	CYCL/00005 4	
RMM.....G0004V00	NS0004	2009/214	Prim	6	CYCL/00005 3	
RMM.....G0004V00	NS0014	2009/215	Prim	5	CYCL/00005 3	
RMM.....G0005V00	NS0005	2009/228	Prim	4	CYCL/00005 2	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	Prim	2	CYCL/00005 1	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.9 CYCLEBY CREATION and DUPLICATE KEEP

In our eighth test, we changed the GDG parmlib option to GDG(CYCLEBY(CREATION) DUPLICATE(KEEP)). Figure 11-42 shows that both versions of the last five generation data sets are retained by the primary VRS definition and that both versions of the oldest generation data sets are retained by the next VRS definition, GDGOPT. However, generation data set 4 will be retained before generation data set 3 because of the creation date.

```
REMOVABLE MEDIA MANAGER                    VITAL RECORDS RETENTION REPORT                                    PAGE          1
    Copyright IBM Corp. 1993,2007          ----- ------- --------- ------                         TIME 14:09:18 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK                 OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF

          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**      SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                                  MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                           FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME

TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0001    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                        12     0
```

Figure 11-42 Result of test case eight

Table 11-11 on page 411 shows that all 12 created data sets are retained. Generation data set G0003V00 is retained before generation data set G0004V00 because of the creation date.

Table 11-11 Details of test case eight

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0001	2009/200	Next	12	**Cycles exceeded**	CYCL/00006 1
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 1
RMM.....G0002V00	NS0002	2009/207	Prim	10	CYCL/00005 5	
RMM.....G0002V00	NS0012	2009/208	Prim	9	CYCL/00005 5	
RMM.....G0003V00	NS0003	2009/221	Prim	6	CYCL/00005 3	
RMM.....G0003V00	NS0013	2009/222	Prim	5	CYCL/00005 3	
RMM.....G0004V00	NS0004	2009/214	Prim	8	CYCL/00005 4	
RMM.....G0004V00	NS0014	2009/215	Prim	7	CYCL/00005 4	
RMM.....G0005V00	NS0005	2009/228	Prim	4	CYCL/00005 2	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 2	
RMM.....G0006V00	NS0006	2009/235	Prim	2	CYCL/00005 1	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.10 CYCLEBY GENERATION and DUPLICATE COUNT

In our ninth test, we changed the GDG parmlib option to GDG(CYCLEBY(GENERATION) DUPLICATE(COUNT)). Figure 11-43 shows that the five newest created data sets are retained by the primary VRS definition. The additional six versions of the next current created generation data sets are retained by the next VRS definition, GDGOPT. The oldest generation data set is dropped by cycles exceeded.

```
REMOVABLE MEDIA MANAGER                  VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007        ----- ------- --------- ------                        TIME 14:09:18 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK                OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF
_____________________________________________________________________________________________________________________________
          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**     SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                                 MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME
_____________________________________________________________________________________________________________________
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                         11     0
```

Figure 11-43 Result of test case six

Table 11-12 on page 412 shows that 11 data sets are retained.

Table 11-12 Details of test case nine

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	**Next**
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Cycles exceeded**
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G0002V00	NS0002	2009/207	Next	10	**Cycles exceeded**	CYCL/00006 5
RMM.....G0002V00	NS0012	2009/208	Next	9	**Cycles exceeded**	CYCL/00006 4
RMM.....G0003V00	NS0003	2009/221	Next	8	**Cycles exceeded**	CYCL/00006 3
RMM.....G0003V00	NS0013	2009/222	Next	7	**Cycles exceeded**	CYCL/00006 2
RMM.....G0004V00	NS0004	2009/214	Next	6	**Cycles exceeded**	CYCL/00006 1
RMM.....G0004V00	NS0014	2009/215	Prim	5	CYCL/00005 5	
RMM.....G0005V00	NS0005	2009/228	Prim	4	CYCL/00005 4	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 3	
RMM.....G0006V00	NS0006	2009/235	Prim	2	CYCL/00005 2	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.5.11 CYCLEBY CREATION and DUPLICATE COUNT

In our tenth test, we changed the GDG parmlib option to GDG(CYCLEBY(GENERATION) DUPLICATE(COUNT)). Figure 11-44 on page 413 shows that the five newest created data sets are retained by the primary VRS definition. An additional six versions of the next current created generation data sets are retained by the next VRS definition, GDGOPT, and the oldest generation data set is dropped by cycles exceeded. Because of the creation date, generation data set 4 will be retained before generation data set 3.

```
REMOVABLE MEDIA MANAGER                  VITAL RECORDS RETENTION REPORT                                   PAGE          1
    Copyright IBM Corp. 1993,2007        ----- ------- --------- ------                        TIME 14:09:18 DATE   2009/223
JOB MASK  DATA SET OR VOLUME MASK                OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE  LASTREF
_____________________________________________________________________________________________________________________________
          RMM.TEST.NEW.GDG.PARMLIB.OPTION.**     SCHLUM   DSN  CYCLES N N   1999/365   0     5     5 HOME           2009/223
          GDGOPT                                 MHLRES7  NEXT CYCLES N N   1999/365         6     6 HOME           2009/223
JOB NAME  DATA SET NAME                          FSEQ  DSEQ  VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY RETDATE    RETNAME
_____________________________________________________________________________________________________________________________
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0016    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0006V00  1     1 NS0006    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0015    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0005V00  1     1 NS0005    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0013    1 SCHLUM   SHELF    LIB2     4800 CYCL/00005 *
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0003V00  1     1 NS0003    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0014    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0004V00  1     1 NS0004    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0012    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0002V00  1     1 NS0002    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
TESTGDG   RMM.TEST.NEW.GDG.PARMLIB.OPTION.G0001V00  1     1 NS0011    1 SCHLUM   SHELF    LIB2     4800 CYCL/00006 GDGOPT
NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                                      11      0
```

Figure 11-44 Result of test case ten

Table 11-13 shows that 11 data sets are retained. Generation data set G0003V00 is retained before generation data set G0004V00 because of the creation date.

Table 11-13 Details of test case ten

DSNAME	VOLSER	CRDATE	VRS	Cycle number	Reason for (non) retention	
					Primary	Next
RMM.....G0001V00	NS0001	2009/200	No	12	**Cycles exceeded**	**Cycles exceeded**
RMM.....G0001V00	NS0011	2009/201	Next	11	**Cycles exceeded**	CYCL/00006 6
RMM.....G0002V00	NS0002	2009/207	Next	10	**Cycles exceeded**	CYCL/00006 5
RMM.....G0002V00	NS0012	2009/208	Next	9	**Cycles exceeded**	CYCL/00006 4
RMM.....G0003V00	NS0003	2009/221	Next	6	**Cycles exceeded**	CYCL/00006 3
RMM.....G0003V00	NS0013	2009/222	Prim	5	CYCL/00005 5	
RMM.....G0004V00	NS0004	2009/214	Next	8	**Cycles exceeded**	CYCL/00006 2
RMM.....G0004V00	NS0014	2009/215	Next	7	**Cycles exceeded**	CYCL/00006 1
RMM.....G0005V00	NS0005	2009/228	Prim	4	CYCL/00005 4	
RMM.....G0005V00	NS0015	2009/229	Prim	3	CYCL/00005 3	
RMM.....G0006V00	NS0006	2009/235	Prim	2	CYCL/00005 2	
RMM.....G0006V00	NS0016	2009/236	Prim	1	CYCL/00005 1	
Next: These data sets are retained by the next VRS definition.						

11.6 Managing DFSMShsm tapes

The interface between DFSMShsm and DFSMSrmm is automatically in place when you install DFSMS, so you do not need to take any action to activate it, or to call it.

You can use the DFSMSrmm PARMLIB OPTION TVEXTPURGE operand to control the processing that DFSMShsm performs when it releases a volume. You can set this volume to pending release or set the volume expiration date to the current date. These are the options for this operand:

NONE DFSMSrmm takes no action for volumes to be purged.

RELEASE DFSMSrmm releases the volume to be purged according to the release actions that are set for the volume. You must run expiration processing to return a volume to scratch status.

Example 11-2 shows adding a VRS to retain DFSMShsm-managed data sets.

Example 11-2 Adding a VRS to retain DFSMShsm-managed data sets

```
RMM ADDVRS DSNAME('mprefix.HMIGTAPE.DATASET') COUNT(99999) CYCLES
RMM ADDVRS DSNAME('mprefix.BACKTAPE.DATASET') COUNT(99999) CYCLES
RMM ADDVRS DSNAME('mprefix.COPY.HMIGTAPE.DATASET') COUNT(99999) CYCLES
RMM ADDVRS DSNAME('mprefix.COPY.BACKTAPE.DATASET') COUNT(99999) CYCLES
```

These VRSs protect all your volumes until DFSMShsm releases the tapes. In this case, the volumes are set to pending release, and, with the next expiration processing, to scratch. Use this option:

- **EXPIRE (*days*)**

  Use the EXPIRE(*days*) option to set the volume expiration date to the current date plus days for volumes to be purged. Use the EXPIRE(*days*) parameter when you use the EXPDT retention method, and use days as a way to delay the expiration of the volume. When using the VRSEL retention method, you can optionally use this operand in combination with VRSs that use the UNTILEXPIRED retention type. First, the EXPIRE(*days*) parameter is applied to set a new volume EXPDT. Next, by running VRSEL, you can optionally extend retention using the extra days retention type. For example, specify EXPIRE(0) when using a VRS with extra days retention, or you can use a nonzero EXPIRE(*days*) value and avoid using an extra days retention VRS.

  The variable ***days*** is the number of days that DFSMSrmm retains the volume before considering it for release. The value is a 1 - 4 digit decimal number and is added to today's date to compute the new expiration date. If the value exceeds the maximum retention period (MAXRETPD), it is reduced to the MAXRETPD value. The default value for days is 0. That is, TVEXTPURGE(EXPIRE) is the same as TVEXTPURGE(EXPIRE(0)).

11.7 DFSMS ACS support

For versions before OS/390 V2R10, you can assign a management value to the data set created on any tape volume. DFSMSrmm calls the EDGUX100 exit at Open/Close/End of Volume (O/C/EOV) time to assign an MV to the data set. During VRSEL processing, the MV assigned to the data set is used to assign a VRS. Figure 11-45 on page 415 shows this process. EDGUX100 also can be used to modify the write to operator (WTO) mount message.

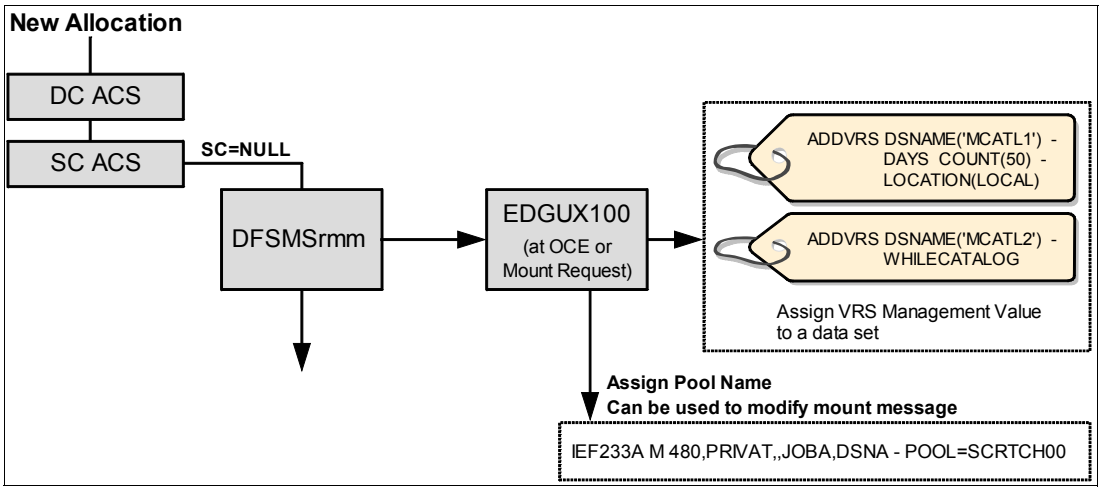


Figure 11-45 Support before OS/390 V2R10 for non-system-managed tapes

Some enhancements were introduced with DFSMSdfp before OS/390 R10 to add pre-ACS support by using the SMS exit IGDACSXT, which gets control before the ACS routines. On OS/390 R10, DFSMSrmm added both pre-ACS support to support the DFSMSdfp processing, and DFSMSrmm ACS support to enable you to use the management class for VRS policy management.

For more details about this function and how to implement this enhancement, see Chapter 12, “DFSMSrmm automatic class selection support” on page 441.

11.8 Modifying VRS

All the modifications to VRS definitions can be made by TSO subcommands or by DFSMSrmm ISPF panels. Here, we use only DFSMSrmm ISPF panels. To see a description of TSO subcommands, see the *DFSMSrmm Guide and Reference,* SC26-7404.

DFSMSrmm does not provide an RMM TSO CHANGEVRS subcommand for modifying an existing VRS. You can use a combination of DELETEVRS and ADDVRS subcommands for modifying an existing VRS. Another way is to use the ISPF dialog.

11.9 Deleting VRS

Before you delete a VRS, you must ensure that you want all the data sets or volumes that are protected by this VRS to be released. You can verify this information in the DFSMSrmm VRSs panel (Figure 11-46 on page 416). Type `M` to the left of the VRS you are planning to delete and press Enter.

```
                          DFSMSrmm VRSs (Page 1 of 3)       Row 1 to 10 of 10
Command ===>                                                  Scroll ===> CSR

The following line commands are valid: C, D, L, M, O, S.
Use the RIGHT command to view other data columns.

S Volume/Data set/Name specification         Job name Type Location Priority
- ------------------------------------------ -------- ---- -------- --------
  *                                                     DSN  HOME     0
  D99000                                                DSN  HOME     0
  M98031                                                DSN  HOME     0
  OPEN                                                  DSN  HOME     0
  SAFE                                                  DSN  REMOTE   0
M YCJRES%.T*.*                                          DSN  HOME     0
  YCJRES1.CAT*.**                                       DSN  HOME     0
```

Figure 11-46 DFSMSrmm VRSs panel

DFSMSrmm shows all the data sets that are matched by this VRS (Figure 11-47) and you can check whether you want to release all of them. The listed data sets might be retained by other more specific VRSs; you can check which VRSs retain a data set by using the `L` line command from the returned list.

```
                               DFSMSrmm Data Sets            Row 1 to 4 of 4
Command ===>                                                 Scroll ===> CSR


The following line commands are valid: C, D, I, L, O, V.
                                            Volume          File Create
S Data set name                             Serial Owner    Seq  Date
- ------------------------------------------ ------ -------- ---- ----------
  YCJRES1.TEST1.VRS2                        TST016 YCJRES1  1    2001/059
  YCJRES1.TEST1.VRS2                        TST023 YCJRES1  1    2001/060
```

Figure 11-47 DFSMSrmm Data Sets panel

To delete the VRS, you can use the ISPF dialog or use the DELETEVRS command.

11.10 DFSMSrmm VRS policy management simplification

In this new release, the VRS processing changed and the RMM TSO subcommand and the reporting are enhanced in the following manner:

- Separation of the data set name mask from the policy
- Release options applied if VRS matched
- Special ABEND, DELETED, and OPEN via DSNAME match
- Find unused VRSs
- Incomplete VRS chains and dummy VRS named *broken*
- Tolerating and removing old functions.
- Conversion to DFSMSrmm from other tape management systems

Important: The DFSMSrmm parmlib option VRSEL(OLD) will be removed in a future release. The suggestion is to migrate from VRSEL(OLD) to VRSEL(NEW) before moving to z/OS V1R8. Each time that you run VRSEL processing and VRSEL(OLD) is in use, the new message EDG2317E and a minimum return code of 4 are issued.

11.10.1 Separation of the data set name mask from the policy

Figure 11-48 shows the structure of a DSNAME VRS and how the concept of separating the data set name mask from the policy itself is put into effect.

The data set name VRS now allows a COUNT of zero, so this first VRS in a chain has no retention specification. The NEXTVRS in the chain and subsequent VRSs now specify the entire policy.

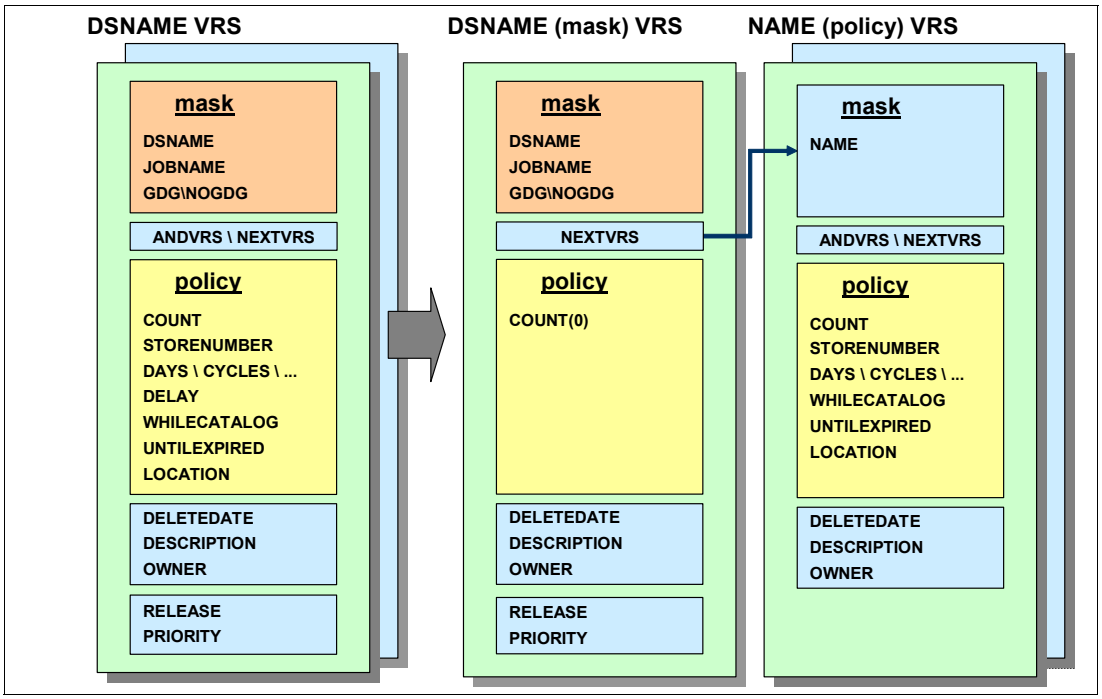


Figure 11-48 DSNAME VRS concept

Setting up DFSMSrmm retention and movement policies can require large numbers of VRSs to be created because each data set name mask VRS also contains the initial retention and movement information.

Separating the data set name mask from the policy itself as shown in Figure 11-49 on page 418 enables clear and well-defined service levels to be set up for tape management.

These policy/service-level VRSs can then be easily modified, as required, without changing the filters that select them.

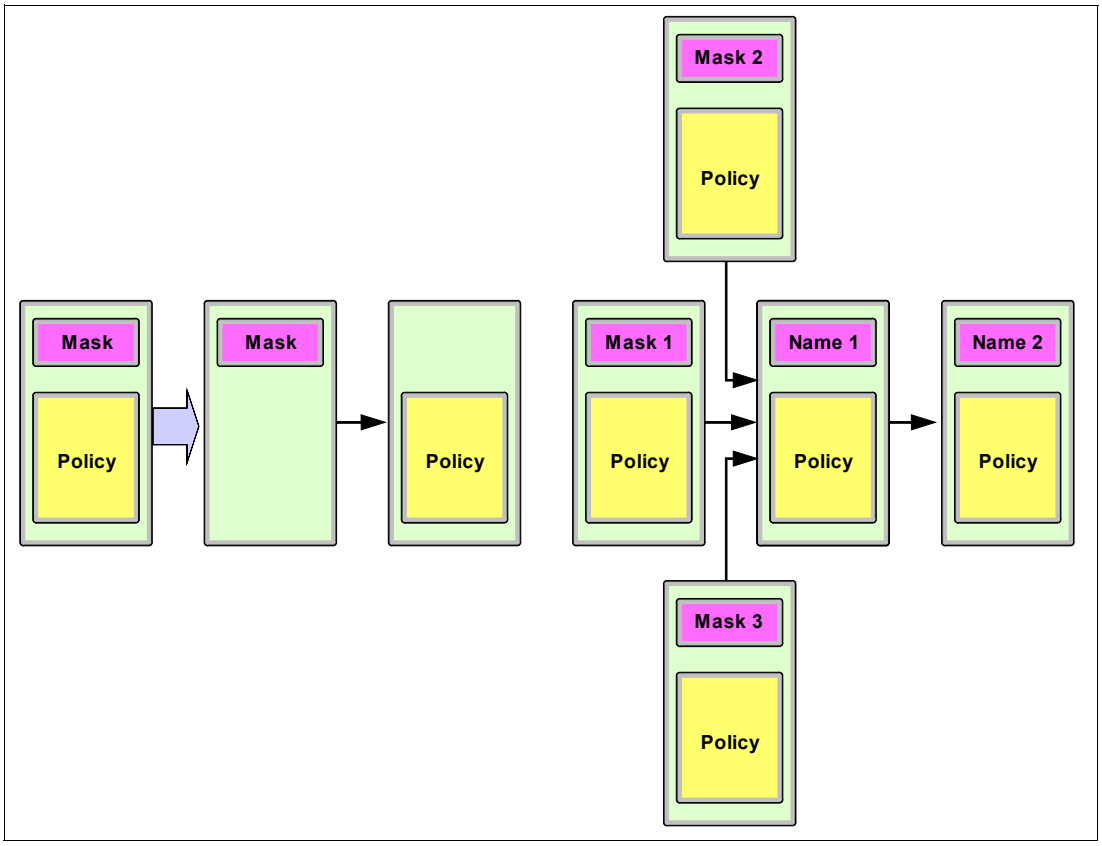


Figure 11-49 Separation of DSNAME mask from the policy

Note: VRSEL(NEW) is required to implement the “Separating the data set name mask from the policy” processing.

If you run DFSMSrmm with VRSEL(OLD), for each VRS where COUNT is set to 0, DFSMSrmm issues the message EDG2225I (see 11.1.9, “VRSEL” on page 379). The processing continues, DFSMSrmm sets a minimum return code of 4, and ignores the VRS; no data sets can match to or use the VRS.

11.10.2 Release options applied if VRS matched

Release options for matching VRSs can be applied regardless of whether the data set is ever VRS retained.

The combination of COUNT(0) and release option changes ensure that a volume can be scratched the same day that a data set on it is created. Figure 11-50 on page 419 shows how you can specify COUNT(0) in a DSNAME VRS definition.

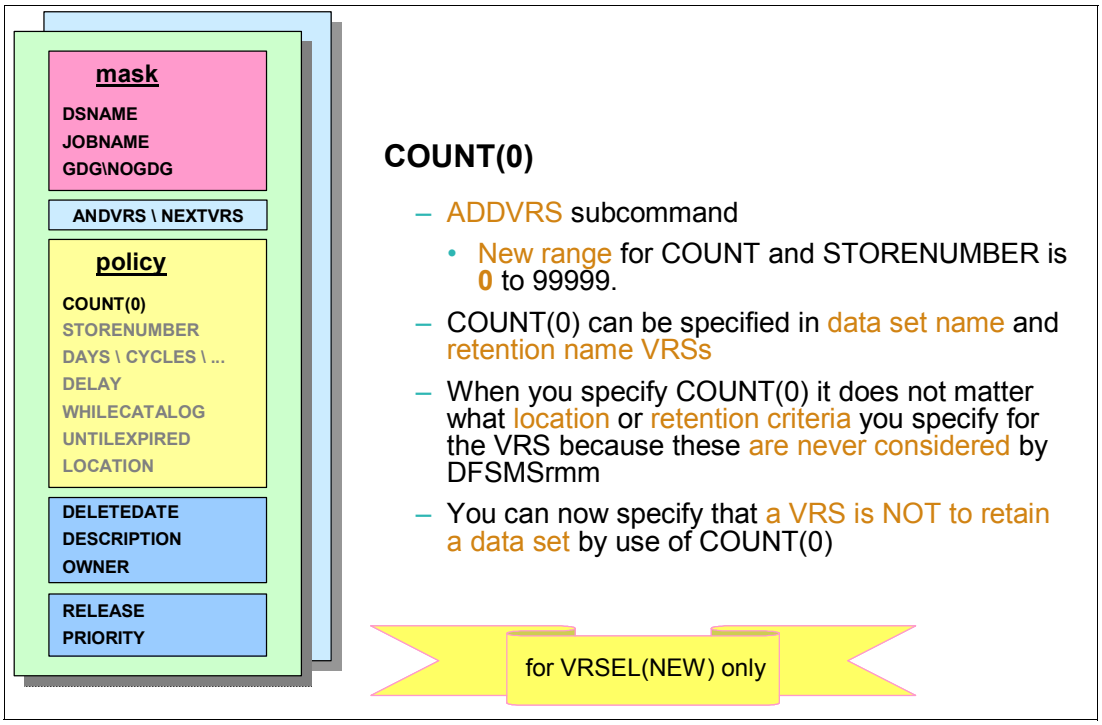


Figure 11-50 VRS count zero processing

When a data set is no longer retained by a VRS, DFSMSrmm releases the volume on which the data set resides only if no data set exists and the volume is retained by a VRS. If you use the DFSMSrmm EDGRMMxx parmlib OPTION command VRSEL(NEW) option and the RMM ADDVRS RELEASE(EXPIRYDATEIGNORE) operand, DFSMSrmm ignores the volume expiration date and uses information in a VRS to control retention. There are two special RELEASE options that are available and you can select one or both of these options:

- EXPIRYDATEIGNORE
- SCRATCHIMMEDIATE

DFSMSrmm does not immediately return a volume to scratch status or to its owner when a volume reaches its expiration date and is not retained by a VRS. You must run expiration processing two times to return a volume to scratch status or to its owner. The first run of expiration processing sets the volume status to pending release. The second run of expiration processing completes the return. Running expiration processing two times gives you time to make changes to the volume status before the volume is released.

Note: Sometimes DFSMSrmm cannot make the return in a single run. For example, there might be other release actions required.

If you do not need to run expiration processing in two runs, specify the DFSMSrmm EDGRMMxx parmlib OPTION command VRSEL(NEW) option and the RMM ADDVRS RELEASE(SCRATCHIMMEDIATE) operand. This enables you to return volumes to scratch in a single run of expiration processing. For more information about the RELEASE option, see the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

The VRS RELEASE option is shown in Figure 11-51 on page 420.

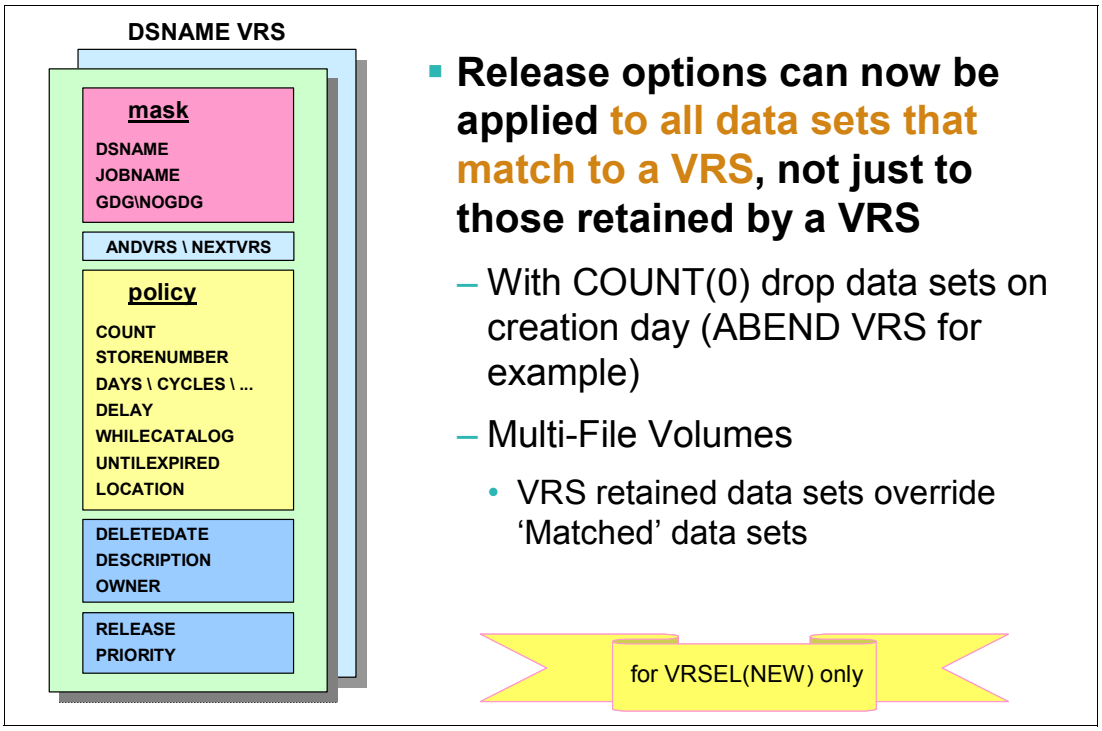


Figure 11-51 VRS RELEASE option

11.10.3 Special ABEND, DELETED, and OPEN via DSNAME match

The support for ABEND and OPEN is extended to allow selection of the appropriate policy using the data set name mask. You can use the reserved data set or job names ABEND and OPEN to specify policies for the following conditions:

- Data sets closed as a result of an abnormal end (ABEND flag in the data set record ON) in a task
- Data sets that are left open (OPEN flag in the volume record ON) or are in use during inventory management

Figure 11-52 shows how you can specify the new reserved ABEND and OPEN masks in the ADDVRS subcommand.

```
▶▶──ADDVRS──────DSNAME(data_set_name_mask)──────────JOBNAME(jobname_mask)──▶
   │         │  │                              │     │                     │
   │──AS─────│  │─DSNAME('ABEND')──────────────│     │─JOBNAME('ABEND')────│
                │                              │     │                     │
                │─DSNAME('OPEN')───────────────│     │─JOBNAME('OPEN')─────│
                │                              │     │                     │
                │─DSNAME('DELETED')────────────│     │─JOBNAME('DELETED')──│
```

Figure 11-52 Add new ABEND or OPEN VRS

This capability allows you to use either data set name masks or job name masks to manage open or abend data sets.

The data set name mask or job name can even be used to match to data sets via a management class or VRS management value as long as the specified data set name mask is not more than a single qualifier.

11.10.4 Find unused VRSs

Now, DFSMSrmm helps you manage your VRSs by VRSEL maintaining date of last reference (DLR) and time of last reference (TLR) for each VRS. This new function also includes the following new functions:

- Counts the number of unused VRSs
- Identifies which VRS policy chains are not being used

You can use this information to identify and delete VRSs that are no longer required.

The VRS last reference date and last reference time are now externalized in the report extract file, in the output of the LISTVRS subcommand. The last reference date is also shown in the REPORT file as shown in Figure 11-53. Also, the REPORT file contains a list of VRS chains that are not used in this VRSEL run.

```
                                   VITAL RECORDS RETENTION REPORT
                                   ----- ------- --------- ------
REMOVABLE MEDIA MANAGER                          VIT                                          PAGE       120
    (C) IBM CORPORATION 1993,2006                ----- ------- --------- ------                TIME 11:21:02 DATE 03/22/2006

 JOB MASK  DATA SET OR VOLUME MASK                    OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE LASTREF
 ---------------------------------------------------------------------------------------------------------------------------
          *.GDG.*.ICRMT.**                            X002     DSN  CYCLES Y N 12/31/1999   0 99999 99999 LOCAL         03/2
 JOB NAME  DATA SET NAME                 2ndVRS   2ndNAMED FSEQ  DSEQ VOLUME VSEQ OWNER    CURRENT  REQUIRED PRTY  RETDATE   RE
 ---------------------------------------------------------------------------------------------------------------------------
 X015IJIC  DT04.GDG.DSNDB06.ICRMT.SYSSTR.G0001V00            12   12 L01699    1 X015     LOCAL    LOCAL     300 WHILECATLG *
 NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                       1     1

 X015IJIC  DT04.GDG.DSNDB06.ICRMT.SYSUSER.G0001V00           11   11 L01699    1 X015     LOCAL    LOCAL     300 WHILECATLG *
 NUMBER OF DATA SETS RETAINED (GROUP STORE) =                                                       1     1

 X015IJIC  DT04.GDG.DSNDB06.ICRMT.SYSVIEWS.G0001V00          13   13 L01699    1 X015     LOCAL    LOCAL     300 WHILECATLG *
 NUMBER OF DATA SETS RETAINED (GROUP STORE) =

                                   UNUSED VRS CHAINS REPORT
                                   ------ --- ------ ------
REMOVABLE MEDIA MANAGER                          UNUS                                         PAGE       121
    (C) IBM CORPORATION 1993,2006                ------ --- ------ ------                TIME 11:21:02 DATE 03/22/2006

 JOB MASK  DATA SET OR VOLUME MASK                    OWNER    TYPE RETN   C X   DELETE   DLY COUNT STNUM LOCATION RLSE LAST
 ---------------------------------------------------------------------------------------------------------------------------
          *.BACKUP.**                                 LIB      DSN  CYCLES Y N 12/31/1999   0 99999 99999 LOCAL         01/2
          *.IMGCPY.IMSDB.**                           LIB      NAME DAYS   N Y 12/31/1999   0 99999 99999 HOME          12/1
          VITAL.**                                    LIB      DSN  CYCLES Y N 12/31/1999   0 99999 99999 CURRENT  IX   05/2
```

LASTREF

03/22/2001

LASTREF

01/23/1992
12/12/2004
05/27/2003

Figure 11-53 Vital Records Retention Report

The MESSAGE file contains an additional message that shows the number of VRSs that are not used by this VRSEL run.

The maintenance of the VRS last reference date and time is supported only if you specified VRSEL(NEW). The following information is listed in the REPORT file LASTREF column:

VRSEL(NEW) It is filled with dates.
VRSEL(OLD) It is no longer supported.

11.10.5 VRS last reference date

After some time, the number of VRSs might grow to a number that is hard to comprehend, especially for no longer used VRSs. Now with DFSMSrmm V1R13, users can list a VRS based on the last reference date.

The SEARCHVRS subcommand is used to create a list of VRSs. Figure 11-54 shows an overview of the SEARCHVRS subcommand with the two new operands, LASTREFDATE and LASTCHANGEDATE. Use the *start_date* operand to specify a date range for your search. A *date_range* consists of a start date and an end date. Each date can be an absolute date in either *yyyy/ddd* or *yyddd* format, or it can be a relative value from which DFSMSrmm calculates the date.

```
▸▸─SEARCHVRS──────────────────────────────────────────────────────────────▸
   SS        |                                | |                          |
             |─LASTCHANGEDATE────────────────| |─LASTREFDATE───────────────|
             |─LASTCHANGEDATE(-date_range-)─| |─LASTREFDATE(-date_range-)─|
               LASTC                            LASTR

  date_range
|──────────────────────────────────────────────────────────────────────────|
  |                          |    |                          |
  |        |─current date─|   |    |       |─current date─|   |
  |─START(-────────────────-)─|    |─END(-────────────────-)─|
    S      |─start_date───|          E    |─end_date─────|
           |─relative─────|                |─relative─────|
```

Figure 11-54 SEARCHVRS subcommand using the LASTREFDATE and LASTCHANGEDATE

The following definitions relate to Figure 11-54:

LASTCHANGEDATE*(date_range)*
Lists the VRSs whose last changed date matches the specified date criteria. The date criteria can be specified using the START and END suboperands.

LASTREFDATE*(date_range)*
Lists the VRSs based on their last reference date. The last reference date of the VRS record is the date of the last inventory management VRSEL run that used this VRS to retain a data set or volume.

date_range
Lists the data sets whose creation date matches the specified date. In the following list, ***cccc*** can be one of these options:
- **LASTCHANGEDATE**
- **LASTREFDATE**

The default is the current date.

***cccc*DATE**
Only data sets whose creation date is the current date are listed.

***cccc*DATE(START)**
Only data sets whose creation date is the current date are listed.

***cccc*DATE(START())**
Only data sets whose creation date is the current date are listed.

***cccc*DATE(START(*start_date*))**
Only data sets whose creation date is on or after the specified

start date are listed, where *start_date* is either an absolute date or a relative date.

***cccc*DATE(END)**
Only data sets whose creation date is the current date are listed.

***cccc*DATE(END())**
Only data sets whose creation date is the current date are listed.

***cccc*DATE(END(*end_date*))**
Only data sets whose creation date is on or before the specified end date are listed, where *end_date* is either an absolute date or a relative date. Because START defaults to the current date, the specified end date is equal to or greater than the current date when START is omitted.

CRDATE(START(*start_date*)END(*end_date*))
Only data sets whose creation date is within the range delimited by the specified start and end dates are listed, where both *start_date* and *end_date* are either an absolute date or a relative date. The specified end date is equal to or greater than the specified start date.

Each of the *start_date* and *end_date* values can be absolute or relative dates.

Absolute dates are specified as either *yyyy/ddd* or *yyddd* format. For example, January 3, 2011 can be specified as 2011/003 or 11003.

Relative Dates are specified as a number of days, months, or years before the current date:

-0
Specifies the current day, current month, and current year

-*n*
Specifies that the date is *n* days before the current date

-*n*M
Specifies that the date is *n* months before the current month and that the current day in the month is the current date

-*n*Y
Specifies that the date is *n* years before the current year and that the current day in the year is the current date

The value range for *n* is 0 - 99999, with a required leading dash (-) and an optional suffix of **M** or **Y**.

Examples are shown to list data sets by creation date:

Today specify SD CRDATE

Three days ago specify SD CRDATE(START(-3) END(-3))

Before January 1, 2000, specifySD CRDATE(START(0000/001) END(1999/365))

On or after January 2, 2005, specify
SD CRDATE(START(2005/002))

RMM ISPF panel updates

Figure 11-55 on page 424 shows the RMM ISPF panel, DFSMSrmm Display Data Set VRS, with the new fields.

```
  Panel  Help
------------------------------------------------------------------------------
                        DFSMSrmm Display Data Set VRS
Command ===>


Data set mask . : 'SCHLUM.RMMTEST.MOVE.**'                        GDG . : NO
     Job name mask . :

Count . . . : 99999                Retention type  . . . . . : CYCLES
                                   While cataloged . . . . . : YES
Delay . . . : 0   Days             Until expired . . . . . . : NO

Location  . . . . . : REMOTE
Number in location  : 1
Priority  . . . . . : 0
                                   Release options:
Next VRS in chain . :              Expiry date ignore  . . . : NO
    Chain using . . :              Scratch immediate . . . . : NO

Owner . . . . . : SCHLUM
Description . . : RETAIN AND MOVE DATA SETS
Last reference  : 2011/326    13:26:39  ( YYYY/DDD  HH:MM:SS )
Delete date . . : 1999/365    ( YYYY/DDD )

Last Change information:
 Date . . . . . . : 2011/326       Time  . : 13:26:40     System  : SC70
 User change date : 2010/289       Time  . : 20:45:17     User ID : MHLRES7
```

Figure 11-55 DFSMSrmm VRS Display Data Set VRS panel

The following definitions apply to Figure 11-55:

Last Reference Display the date and time of the last inventory management VRSEL run that used the VRS to retain a data set or volume.

Last change information
Display details of the most recent change to the record:

- **Date** Displays the last change date.
- **Time** Displays the last change time.
- **System** Displays the ID of the system on which the last change occurred.
- **User change date** Displays the date of the most recent change by a user, other than by a DFSMSrmm internal function.
- **User change time** Displays the time of the most recent change by a user, other than by a DFSMSrmm internal function.
- **User ID** Displays the ID of the user who caused the most recent change. If the most recent change was made by DFSMSrmm processing, the ID starts with an asterisk (*). Internal IDs include these values:
  - ***OAM** DFSMSrmm system-managed tape support
  - ***HKP** Inventory management
  - ***OCE** DFSMSrmm OPEN/CLOSE/EOV support

11.10.6 Incomplete VRS chains and dummy VRS named *broken*

If VRSEL processing finds an incomplete chain, it is reported with EDG2230I, but the processing continues. In this case, DFSMSrmm sets a minimum return code of 4 and any data set that matches this incomplete chain is retained by a special Name VRS of *broken*. Figure 11-56 shows an example of an incomplete VRS chain.

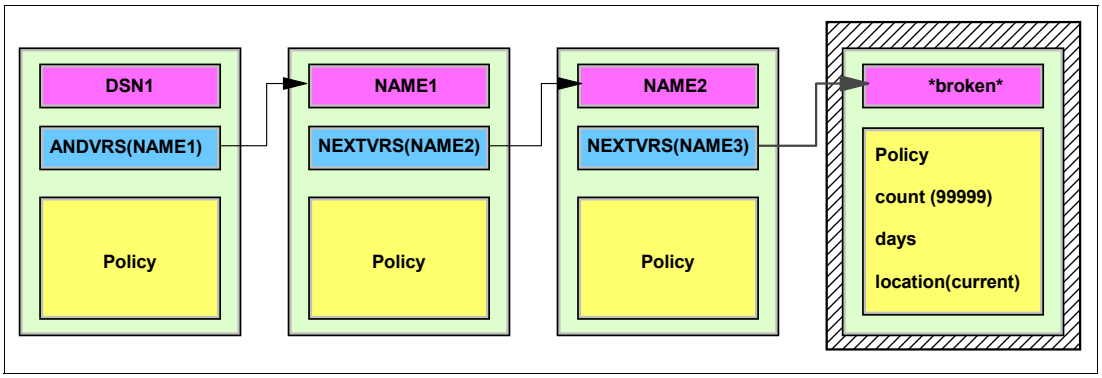


Figure 11-56 Incomplete VRS chain

Two new messages are described that might display if the VRSEL processing detects errors:

EDG2230I `NEXTVRS` *`name_vrs`* `DOES NOT EXIST. CHAINING` *`vrs_type`* `VRS IS vrs_mask.`

The following variables apply to this message:

vrs_name The NAME VRS defined by NEXTVRS.

vrs_type DSN - DSNAME VRS.
VOL - VOLUME VRS.
NAME - NAME VRS.

vrs_mask The mask that uniquely identifies the VRS with the chaining error. For DSNAME VRSs, the mask includes the data set name and, optionally, the job name.

Severity Information.

Explanation During inventory management vital record processing, DFSMSrmm checks all VRS chains by following the chain using the NEXTVRS values. The VRS displayed in the message does not exist in the DFSMSrmm control data set.

Source DFSMSrmm.

Detecting module EDGVREC0.

System action Processing continues, and DFSMSrmm sets a minimum return code of 4. For VRSEL(OLD), DFSMSrmm retains additional data sets or volumes, in the home location, up to the COUNT value specified in the initial VRS in the chain. For VRSEL(NEW), additional data sets are retained in the current location, permanently.

Operator response None.

System programmer response

Add the missing VRS or correct the NEXTVRS value specified on the VRS displayed in the message.

Routing codes N/A.

Descriptor codes None.

EDG2317E `MIGRATION FROM VRSEL(OLD) TO VRSEL(NEW) IS RECOMMENDED.`

The following descriptions apply to this message:

Severity Warning.

Explanation During inventory management vital record processing, DFSMSrmm checks the VRSEL option that you defined in the EDGRMMxx parmlib. The VRSEL(OLD) option will be removed in a future release of z/OS. You need to migrate to using VRSEL(NEW).

Source DFSMSrmm.

Detecting module EDGMHKP.

System action Processing continues. A minimum return code of 4 is set.

Operator response None.

System programmer response

Plan a migration to VRSEL(NEW). See the migration planning steps that are documented in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

Routing codes 11.

Descriptor codes None.

11.10.7 Tolerating and removing old functions

Perform a migration to VRSEL(NEW). See the migration planning steps that are documented in the *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405.

The use of the old VRS operands, STARTNUMBER, LOCATION(BOTH) and STORENUMBER(*xx,yy*), is prevented.

The toleration APAR OA13355 includes a ++HOLD(ACTION) and requires that VRSs are cleaned up before you are able to run EDGHSKP with VRSEL on a z/OS V1R8 system.

The cleanup actions are documented under message EDG2221E. EDG2221E sets return code 12 instead of 4.

Important: As long as you do not implement COUNT(0) or JOBNAME(ABEND\OPEN), you can run VRSEL processing on either a toleration system or z/OS V1R8.

Error messages

This new message is shown if the VRSEL processing detects errors:

EDG2222E `Type VRS FOR mask SPECIFIES UNSUPPORTED OPTIONS - SOME RETENTION OPTIONS IGNORED.`

The following variables apply to this message:

type The type of VRS. The type is either DSN - DSNAME type VRS or VOL - VOLUME type VRS.

mask The *mask* is the VRS data set name or volume serial number.

The following descriptions apply to this message:

Severity Information.

Explanation During vital record processing, DFSMSrmm found a VRS that contains unsupported options. The unsupported options are STARTNUMBER and LOCATION(BOTH).

Source DFSMSrmm.

Detecting module EDGVREC0.

System action Processing ends. A return code of 12 is set.

Operator response None.

System programmer response

You must replace the VRS with other VRSs that provide the retention options that you require. For example, if you use LOCATION(BOTH), you can replace it with the use of the NEXTVRS operand. This example shows first an unsupported VRS and then the equivalent supported VRSs that you might use.

Unsupported:

```
RMM ADDVRS DSNAME(data_set_name_mask) -
       CYCLES COUNT(5) LOCATION(BOTH) -
       STORENUMBER(2,1)
```

Supported:

```
RMM ADDVRS DSNAME(data_set_name_mask) -
      CYCLES COUNT(5) LOCATION(LOCAL) -
      STORENUMBER(2) NEXTVRS(DIST1C)
RMM ADDVRS NAME(DIST1C) LOCATION(DISTANT) -
      STORENUMBER(1)
```

If you use STARTNUMBER, you can replace it with the use of the NEXTVRS operand. This example shows first an unsupported VRS and then the equivalent supported VRSs that you might use.

Unsupported:

```
RMM ADDVRS DSNAME(data_set_name_mask) -
       CYCLES COUNT(3) LOCATION(VAULT1) -
       STORENUMBER(2) STARTNUMBER(1)
```

Supported:

```
RMM ADDVRS DSNAME(data_set_name_mask) -
       CYCLES COUNT(3) LOCATION(HOME) -
       STORENUMBER(1) NEXTVRS(VLT12C)
RMM ADDVRS NAME(VLT12C) LOCATION(VAULT1) -
       STORENUMBER(2)
```

In addition, IBM suggests that you perform a migration to VRSEL(NEW). See the migration planning steps that are documented in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

Routing code 11.
Descriptor code 7.

11.10.8 Conversion to DFSMSrmm from other tape management systems

Before release V1R8, DFSMSrmm converted only DSN, LABEL, and JOB information to UXTABLE entries. Now, it also supports MGMTCLAS and ABEND keywords as shown in Figure 11-57 on page 428.

```
{                   }{              {keyword       }}
{                   }{              {preferred date}}[,J=create.jobname ]
{D=dsname[-]        }{,LABEL=EXPDT={Julian date    }}[,JOB=create.jobname]
{DSN=dsname[-]      }{,LABEL=RETPD=nnnn             }[,JOBNAME=create.jobname]
 MGMTCLAS=smsclass  {,LABEL=WRETPD=nnnn             }[,SELECT=ALL            ]
 M=smsclass         {,ABEND=RETPD=nnnn              }
                    {,ABEND=WRETPD=nnnn             }
                    {,ABEND=EXPDT={keyword         }}
                                  {preferred date}
                                  {Julian date   }
```

Figure 11-57 Support for MGMTCLAS and ABEND retention data set (RDS) keywords

EDGCRFMT and EDGRSRDS conversion programs

If you have a CA-1 retention data set (RDS) entry, this RDS entry is converted in two steps to create a UXTABLE entry that can be used in the EDGUX100 user exit:

1. EDGCRFMT creates input for EDGCSRDS.
2. EDGCSRDS builds the UXTABLE entries.

Example 11-3 shows some CA-1RDS entries.

Example 11-3 CA-1 RDS entries

```
M=SMSCLAS1,LABEL=RETPD=2010/011,JOB=TEST
MGMTCLAS=SMSCLAS2,LABEL=RETPD=2020/011
DSN=RXXXX.XXXX,ABEND=RETPD=2010/001,JOB=TESTTEST
DSN=RXXXX.XXXX,LABEL=RETPD=2010/001,JOB=TESTTEST
DSN=R3650.*,LABEL=RETPD=3650,JOB=COMBIN*
```

The CA-1 RDS entries are converted to UXTABLE as shown in Figure 11-58 on page 429. For more information about a CA-1 to DFSMSrmm conversion, see *Converting to DFSMSrmm from CA-1*, SG24-6241. This publication describes the conversion in detail.

```
* start of RDS entries
         EDGCVRSG DSN=RXXXX.XXXX.G%%%%V%%,  X
               JOB=ABEND,                   X
               RO=001,                      X
               RETPD=2010
         EDGCVRSG DSN=RXXXX.XXXX,           X
               JOB=ABEND,                   X
               RO=001,                      X
               RETPD=2010
         EDGCVRSG DSN=RXXXX.XXXX.G%%%%V%%,  X
               JOB=TESTTEST,                X
               RO=001,                      X
               RETPD=2010
         EDGCVRSG DSN=RXXXX.XXXX,           X
               JOB=TESTTEST,                X
               RO=001,                      X
               RETPD=2010
         EDGCVRSG DSN=R3650.*,              X
               JOB=COMBIN*,                 X
               RO=NO,                       X
               RETPD=3650
         EDGCVRSG DSN=SMSCLAS1,             X
               JOB=TEST,                    X
               RO=011,                      X
               RETPD=2010
         EDGCVRSG DSN=SMSCLAS2,             X
               RO=011,                      X
               RETPD=2020
* default RP value
         EDGCVRSG DSN=*,                    X
               RO=NO,                       X
               RETPD=3
```

Figure 11-58 Sample UXTABLE entries

EDGCSVDS conversion program

EDGCSVDS converts the vaulting policies from the vault pattern description data set (VPDD) into K records representing the DFSMSrmm storage location movement policies.

Processing is changed to implement data set filter VRSs for all data set name (DSN) entries found in the VPDD. Instead of storing data set and job names in memory until the vault statements are reached, and then looping through them to create K records, we can now create a K record for the filter as we read the entry from the VPDD. This way, we no longer have memory limitations in EDGCSVDS.

EDGCNVT conversion program

EDGCNVT is updated to allow COUNT(0) in the count that is filed on the EDGCKREC record. This program converts the output of the data extraction programs to DFSMSrmm format CDS records or ADDVRS commands as shown in Figure 11-59 on page 430.

Separating data set name mask from the policy itself

before V1R8

```
RMM ADDVRS DSN('A1.TEST                                        ') NOGDG
           CYCLES      COUNT(     1) DELAY( 0)
           OWNER(K       ) DELETE(1999/365)
           LOCATION(MINSK    ) STORENUM(    1)  PRIORITY(   0)
RMM ADDVRS DSN('A2.TEST                                        ') NOGDG
           CYCLES      COUNT(     1) DELAY( 0)
           OWNER(K       ) DELETE(1999/365)
           LOCATION(MINSK    ) STORENUM(    1)  PRIORITY(   0)
```

VPD

```
DSN=A1.TEST
DSN=A2.TEST
V=MINSK,C=1
```

now

```
RMM ADDVRS DSN('A1.TEST                                        ') NOGDG
           CYCLES      COUNT(     0) DELAY( 0)
           OWNER(K       ) DELETE(1999/365)
           LOCATION(CURRENT ) STORENUM(    0)  PRIORITY(   0)
           NEXTVRS(VRS00001)
RMM ADDVRS NAME(VRS00001)  CYCLES      COUNT(     1)
           OWNER(K       ) DELETE(1999/365)
           LOCATION(MINSK    ) STORENUM(    1)
           NEXTVRS(VRS00002)
RMM ADDVRS DSN('A2.TEST                                        ') NOGDG
           CYCLES      COUNT(     0) DELAY( 0)
           OWNER(K       ) DELETE(1999/365)
           LOCATION(CURRENT ) STORENUM(    0)  PRIORITY(   0)
           NEXTVRS(VRS00001)
```

Figure 11-59 EDGCNVT RMM TSO ADDVRS commands

11.11 Trial run

DFSMSrmm provides the EDGHSKP utility to help you perform inventory management. When you specify the VERIFY parameter, DFSMSrmm performs a trial run of vital record processing, so you can see how the vital record processing changes affect the data set and volume information before the CDS is updated. Figure 11-60 contains a sample JCL for inventory management trial run processing.

```
//HSKP EXEC PGM=EDGHSKP,PARM='VRSEL,VERIFY,DATE(+5)'
//MESSAGE   DD  DISP=SHR,DSN=HSKP.MESSAGES
//REPORT    DD  DISP=SHR,DSN=HSKP.VRS.REPORT
//ACTIVITY  DD  DISP=SHR,DSN=HSKP.ACTIVITY
```

Figure 11-60 Inventory management trial run

The parameter DATE(+5) adds five days to the current date to perform a trial run on this date.

The activity file is not intended to be a report. It merely contains information about changes made to data sets during vital record processing. DFSMSrmm provides a sample job EDGJACTP in SAMPLIB that shows how to selectively format and print fields.

The output for this job consists of thirteen data sets that contain the following reports:

RUNINFO	Controlling options and parameters
VRS	Data sets with changed vital record status
VRSS	Summary by status
RETDATE	Data sets with new retention dates
RETDS	Summary by new date
MATCHVRS	Data sets with new matching VRS
MATCHVS	Summary by new matching VRS
SUBCHN	Data sets retained by new part of chain
SUBCHNS	Summary by new retaining subchain
EXPDROP	EXPDT-retained volumes for EXPDTDROP

EXPDROPS	Summary of EXPDT-retained volumes
VRSRETN	Newly assigned volumes for VRSRETAIN
VRSRETNS	Summary of newly assigned volumes for VRSRETAIN

For more information about these reports and the EDGJACTP job, see *DFSMSrmm Reporting,* SC26-7406.

11.12 VRS examples

The following examples show a VRS chain and subchain to explain the difference between these concepts.

11.12.1 ADDVRS operand defaults

In the examples, we use the default values for the operands of the ADDVRS subcommand:

COUNT	99999
DELAY	0
DELETEDATE	1999/365
LOCATION	HOME
PRIORITY	0
STORENUMBER	99999

11.12.2 Adding a single VRS

In the first example (see Figure 11-61), we use a single VRS to retain and move a data set. The data sets shown in Figure 11-62 on page 432 are retained only by this PRIMARY VRS.

This DSN VRS has the following tasks:

- Retain ten cycles of each matching data set group.
- All instances of a data set created on a single calendar day are considered to be a single cycle.
- The newest five cycles must be stored in the outside storage location SAVE, and all others in their HOME location.

Primary DSN VRS

```
RMM ADDVRS                -
     DSN('MARY.**')       -
     BYDAYSCYCLES         -
     COUNT(10)            -
     STORENUMBER(5)       -
     LOCATION(SAVE)
```

Figure 11-61 Single VRS example

Figure 11-62 on page 432 shows the data sets and the location that is retained by the single VRS definition.

```
Data set name          Creation     Location  Creation
                       date                   occurrence
MARY.TSO.DATA.BACKUP   2002/123      HOME     -9
MARY.TSO.DATA.BACKUP   2002/124      HOME     -8
MARY.TSO.DATA.BACKUP   2002/125      HOME     -7
MARY.TSO.DATA.BACKUP   2002/126      HOME     -6
MARY.TSO.DATA.BACKUP   2002/127      HOME     -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/127      HOME     -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/130      SAVE     -4
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/132      SAVE     -2
MARY.TSO.DATA.BACKUP   2002/133      SAVE     -1
MARY.TSO.DATA.BACKUP   2002/134      SAVE      0
```

Figure 11-62 Single VRS retention and location results

11.12.3 VRS chain with one NEXTVRS operand

In the second example (Figure 11-63), we use the NEXTVRS operand to have an additional LOCATION definition. The request for this VRS is almost the same as the request in Figure 11-61 on page 431. The difference is that the newest five cycles must always be in the HOME location.

This VRS specification has the following tasks:

- Retain ten cycles of each matching data set group.
- All instances of a data set that are created on a single calendar day are considered to be a single cycle.
- The newest cycle (0) must be stored in the HOME location. You do not need to specify the LOCATION(HOME) operand, because HOME is the default.
- The next five cycles (-1 to -5) must be stored in the outside storage location SAVE.
- All other cycles (-6 to -9) must be retained in the HOME location.

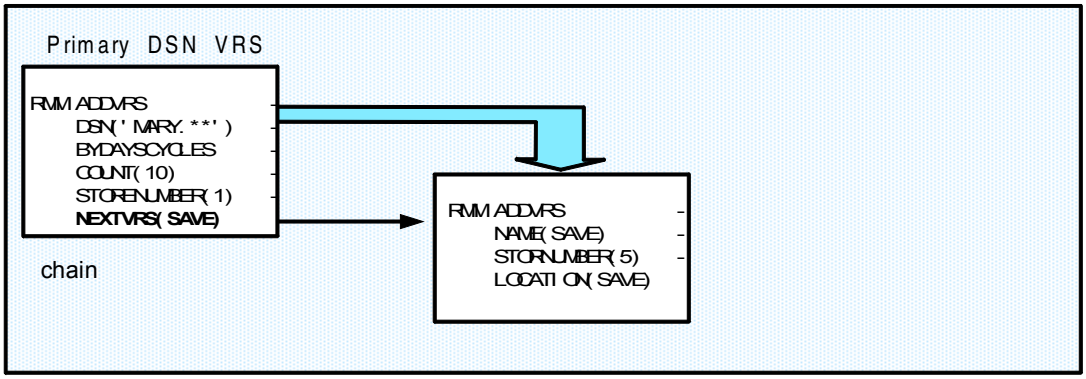


Figure 11-63 VRS chain with one NEXTVRS operand

Figure 11-64 on page 433 shows the data sets and their location retained by this VRS definition.

```
Data set name          Creation     Location  Creation
                       date                   occurrence
MARY.TSO.DATA.BACKUP   2002/123      HOME     -9
MARY.TSO.DATA.BACKUP   2002/124      HOME     -8
MARY.TSO.DATA.BACKUP   2002/125      HOME     -7
MARY.TSO.DATA.BACKUP   2002/126      HOME     -6
MARY.TSO.DATA.BACKUP   2002/127      SAVE     -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/127      SAVE     -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/130      SAVE     -4
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/132      SAVE     -2
MARY.TSO.DATA.BACKUP   2002/133      SAVE     -1
MARY.TSO.DATA.BACKUP   2002/134      HOME      0
```

Figure 11-64 Single VRS using the NEXTVRS operand results

11.12.4 VRS chain with two NEXTVRS definitions

The third example (see Figure 11-65) shows a VRS with two NEXTVRS definitions that have more than one outside LOCATION definition. The only difference between the first example and the second example is that the data sets must be moved to an auxiliary storage location.

This VRS specification has the following tasks:

- Retain ten cycles of each matching data set group.
- All instances of a data set created on a single calendar day are considered to be a single cycle.
- The newest cycles (0) must be stored in the HOME location. You do not need to specify the LOCATION(HOME) operand, because HOME is the default.
- The next three cycles (-1 to -3) must be stored in the outside storage location SAVE.
- The cycles (-4 to -6) must be stored in the outside storage location MUNICH.
- All other cycles (-7 to -9) must be retained in the HOME location.

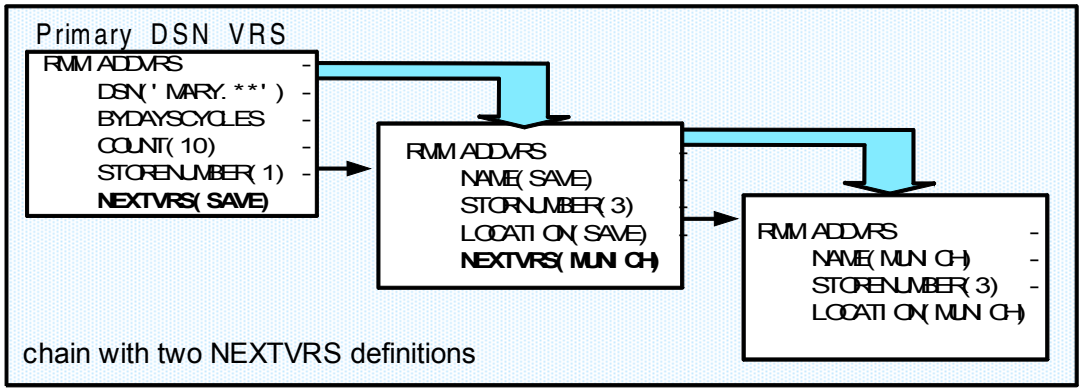


Figure 11-65 VRS chain with two NEXTVRS operands

Figure 11-66 shows the data sets and their location retained by this VRS definition.

```
Data set name          Creation     Location  Creation
                       date                   occurrence
MARY.TSO.DATA.BACKUP   2002/123      HOME     -9
MARY.TSO.DATA.BACKUP   2002/124      HOME     -8
MARY.TSO.DATA.BACKUP   2002/125      HOME     -7
MARY.TSO.DATA.BACKUP   2002/126      MUNICH   -6
MARY.TSO.DATA.BACKUP   2002/127      MUNICH   -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/127      MUNICH   -5  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/130      MUNICH   -4
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/131      SAVE     -3  BYDAYSCYCLE
MARY.TSO.DATA.BACKUP   2002/132      SAVE     -2
MARY.TSO.DATA.BACKUP   2002/133      SAVE     -1
MARY.TSO.DATA.BACKUP   2002/134      HOME      0
```

Figure 11-66 Single VRS using two NEXTVRS operand results

11.12.5 VRS chain with two NEXTVRS operands with retention information

Figure 11-67 shows a management class (MC) or a management value (MV) VRS using two NEXTVRS operands to have more than one retention criterion. All data sets that are retained by this VRS are retained in the HOME location. This example is a VRS chain with two additional subchains.

This VRS specification has the following tasks:

- Retain a data set for five calendar days by specifying the operand DAYS.
- After five calendar days, retain the data set as long as it is cataloged by specifying the WHILECATALOG operand.
- If the data set is no longer cataloged, retain the data set for one additional day by specifying the EXTRADAYS operand.

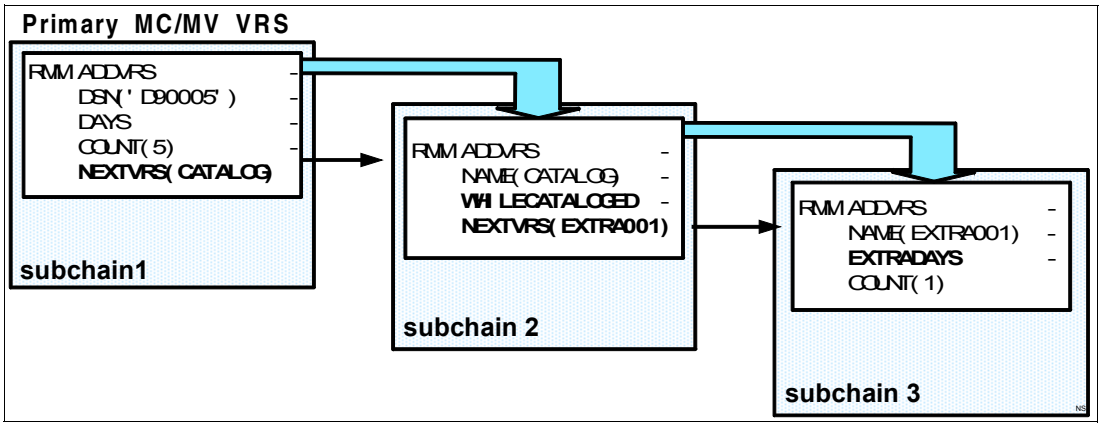


Figure 11-67 VRS chain with retention information example

In Figure 11-68 on page 435, the generation data group (GDG) is defined with a limit of 10. Generation data set MARY.TSO.DATA.G0011V00 was uncataloged manually yesterday, and the generation data set MARY.TSO.DATA.G0010V00 was uncataloged manually last week. The first of the two MARY.TSO.DATA.G0013V00 generation data sets was created incorrectly and not cataloged.

```
Data set name           Creation     Location  Data set   Retention
                        date                   cataloged  criteria
MARY.TSO.DATA.G0009V00  2002/123      HOME        Y       WHILECATALOGED
MARY.TSO.DATA.G0010V00  2002/124      HOME        N       not retained
MARY.TSO.DATA.G0011V00  2002/125      HOME        N       EXTRADAYS
MARY.TSO.DATA.G0012V00  2002/126      MUNICH      Y       WHILECATALOGED
MARY.TSO.DATA.G0013V00  2002/127      MUNICH      N       not retained
MARY.TSO.DATA.G0013V00  2002/127      MUNICH      Y       WHILECATALOGED
MARY.TSO.DATA.G0014V00  2002/130      MUNICH      Y       DAYS
MARY.TSO.DATA.G0015V00  2002/131      SAVE        Y       DAYS
MARY.TSO.DATA.G0015V00  2002/131      SAVE        Y       DAYS
MARY.TSO.DATA.G0016V00  2002/132      SAVE        Y       DAYS
MARY.TSO.DATA.G0017V00  2002/133      SAVE        Y       DAYS
MARY.TSO.DATA.G0018V00  2002/134      HOME        Y       DAYS
```

Figure 11-68 Results of a single VRS using two retention NEXTVRS operands

11.12.6 VRS chain with two ANDVRS operands

The fifth example (see Figure 11-69) shows a complex VRS with two ANDVRS operands to have more than one retention criterion for retaining a data set. If you specify ANDVRS, all definitions must be true to retain the data set. If any definition is not true, the data set is no longer retained by this VRS chain.

This VRS has the following tasks:

- Retain a data set for three cycles.
- Only retain the data set if the data set has been used in the last six days.
- The data set must be moved to the outside storage location STOREX until the volume expiration date is reached.

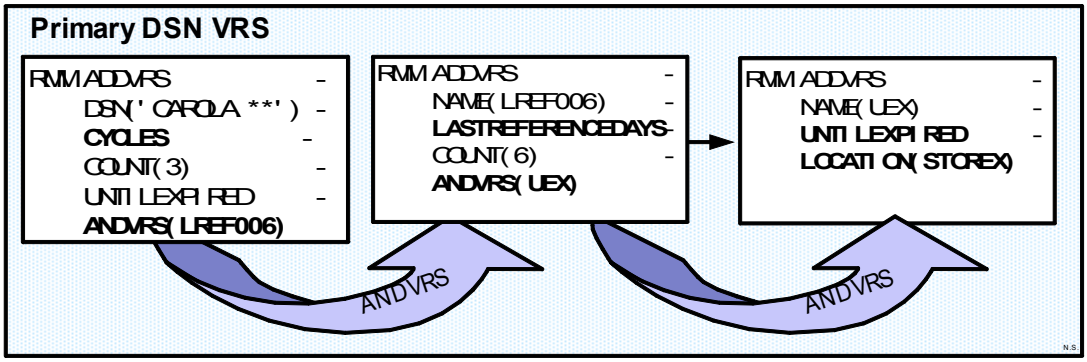


Figure 11-69 VRS chain with two ANDVRS operands example

Figure 11-70 on page 436 shows the data sets and their location retained by the VRS definition in Figure 11-69. Only the generation data sets G0016V00, G0017V00, and G0018V00 are retained by the VRS because we specified CYCLES COUNT(3).

```
Data set name           Creation  Exp.       Last used    Retention
                        date      date       date         criteria
                                                          WC  LR  UEX
MARY.TSO.DATA.G0009V00  2002/123  2002/123   2002/123     N   N   N
MARY.TSO.DATA.G0010V00  2002/124  2002/124   2002/124     N   N   N
MARY.TSO.DATA.G0011V00  2002/125  2002/125   2002/231     N   N   N
MARY.TSO.DATA.G0012V00  2002/126  2002/126   2002/126     N   N   Y
MARY.TSO.DATA.G0013V00  2002/127  2002/127   2002/127     N   N   Y
MARY.TSO.DATA.G0013V00  2002/127  2002/127   2002/132     N   Y   Y
MARY.TSO.DATA.G0014V00  2002/130  2002/130   2002/130     N   N   Y
MARY.TSO.DATA.G0015V00  2002/131  2002/131   2002/100     N   Y   Y
MARY.TSO.DATA.G0015V00  2002/131  2002/131   2002/131     N   N   Y
MARY.TSO.DATA.G0016V00  2002/132  2002/132   2002/132     Y   Y   Y <===
MARY.TSO.DATA.G0017V00  2002/133  2002/133   2002/133     Y   Y   Y <===
MARY.TSO.DATA.G0018V00  2002/134  2002/134   2002/134     Y   Y   Y <===
```

Figure 11-70 Results of a single VRS using two retention NEXTVRS operands

11.12.7 VRS chain with two subchains

Figure 11-71 shows a complex VRS with an ANDVRS definition, and an operand in a VRS or just NEXTVRS operand with retention information. This VRS has the following tasks:

- Retain a data set for three cycles.
- Only retain the data set if the data set has been used in the last six calendar days.
- If the data set is no longer retained by the first subchain, the data set must be retained for five additional days.

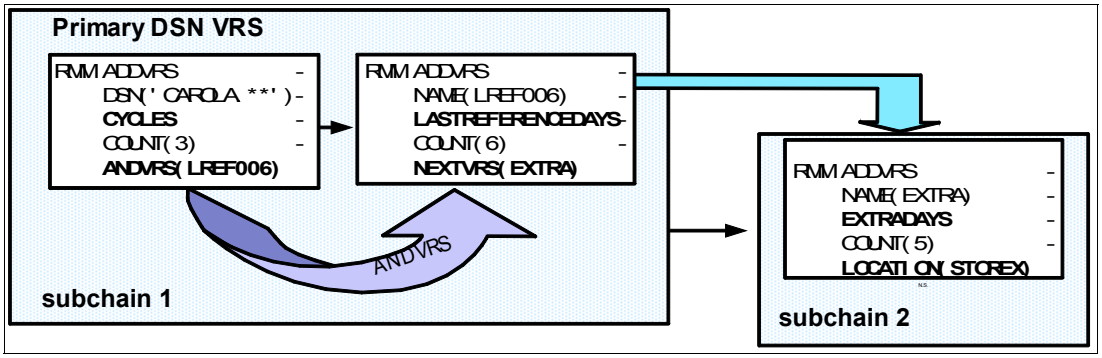


Figure 11-71 VRS chain with two subchains example

Figure 11-72 on page 437 shows the data sets and their location retained by the VRS definition. The generation data sets G0016V00, G0017V00, and G0018V00 are retained by the first subchain, and the generation data set G0015V00 is retained and moved to the storage location STOREX by the secondary subchain.

```
Data set name           Creation  Exp.       Last used    Retention
                        date      date       date         criteria
                                                          WC  LR       *
MARY.TSO.DATA.G0009V00  2002/123  2002/123   2002/123     N   N
MARY.TSO.DATA.G0010V00  2002/124  2002/124   2002/124     N   N
MARY.TSO.DATA.G0011V00  2002/125  2002/125   2002/231     N   N
MARY.TSO.DATA.G0012V00  2002/126  2002/126   2002/126     N   N
MARY.TSO.DATA.G0013V00  2002/127  2002/127   2002/127     N   N
MARY.TSO.DATA.G0013V00  2002/127  2002/127   2002/132     N   N
MARY.TSO.DATA.G0014V00  2002/130  2002/130   2002/130     N   N
MARY.TSO.DATA.G0015V00  2002/131  2002/131   2002/100     N   N
MARY.TSO.DATA.G0015V00  2002/131  2002/131   2002/131     N   Y   <=== 2
MARY.TSO.DATA.G0016V00  2002/132  2002/132   2002/132     Y   Y   <=== 1
MARY.TSO.DATA.G0017V00  2002/133  2002/133   2002/133     Y   Y   <=== 1
MARY.TSO.DATA.G0018V00  2002/134  2002/134   2002/134     Y   Y   <=== 1

* = retained by VRS subchain
```

Figure 11-72 ANDVRS and NEXTVRS with retention operand results

11.13 Complex VRS chain with three subchains

Figure 11-73 on page 438 shows a complex VRS with a primary DSN VRS that has two NEXTVRS definitions followed by two subchains with retention information. The second subchain has a NEXTVRS definition, and the third subchain uses the ANDVRS operand.

This VRS has the following tasks:

1. The newest 20 cycles are retained by the first subchain:
   - The newest five cycles are always in the HOME location.
   - The next five cycles are moved to the storage location SAVE.
   - The cycles -10 to -14 are retained in the storage location MUNICH.
   - The last five cycles are retained in the HOME location.
2. After the data sets are no longer retained by the first subchain, we retain all data sets for 365 calendar days since creation:
   - The first 60 days, retain the data sets in the HOME location.
   - Then, retain the data sets for 14 days in the storage location MAINZ.
   - The last 291 days, retain the data sets in the HOME location.
3. The third subchain retains the data set only if we have up to 30 cycles and if the data set has been read or written to in the last 400 calendar days.

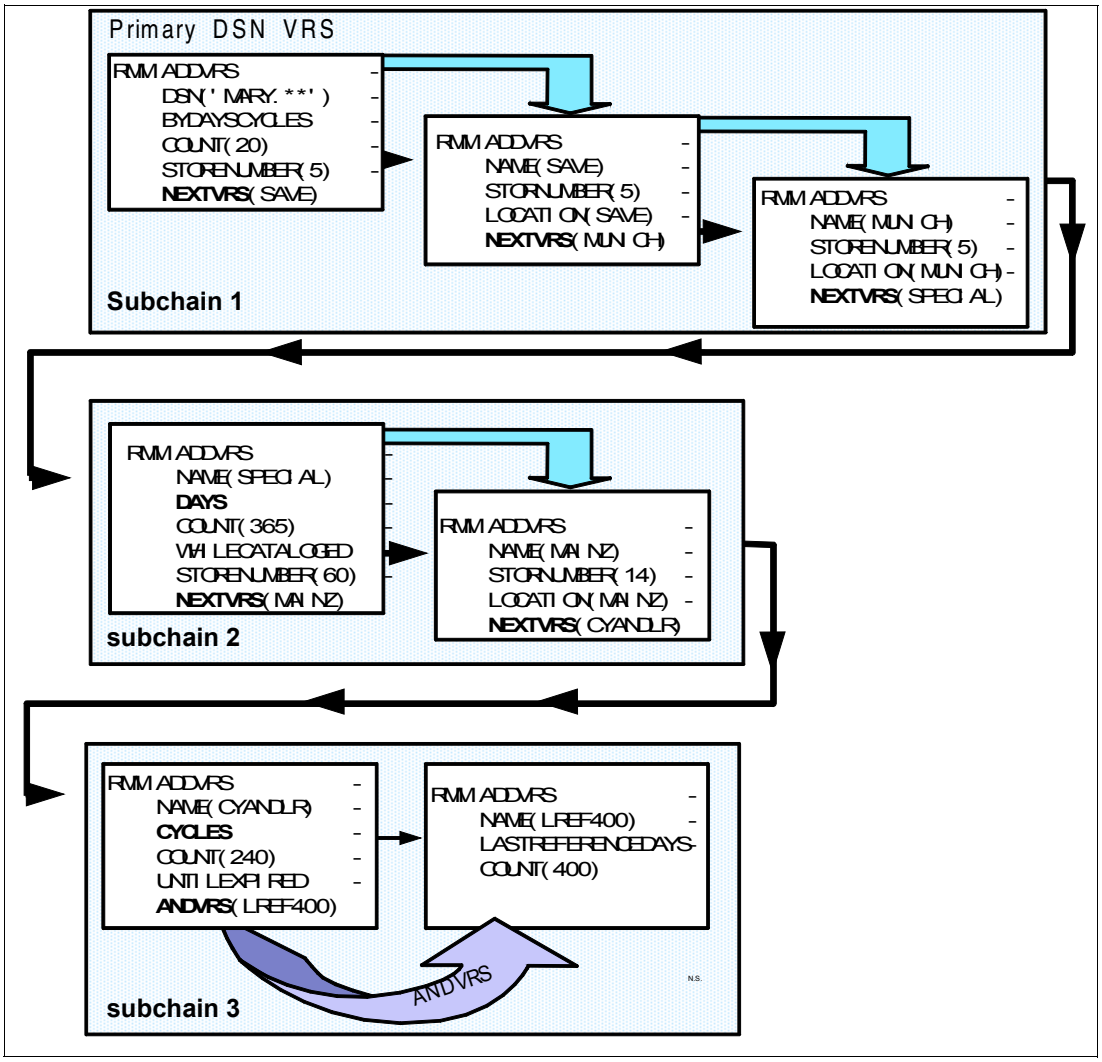


Figure 11-73 VRS chain with three subchains example

11.14 VRS hints and tips

With VRSEL(NEW) parameters, system-managed and non-system-managed tape data sets can be managed using ACS routines. This allows you to simplify VRS specifications and centralize the data set specifications in the ACS routines.

You can benefit from the VRS support introduced by VRSEL(NEW) if you follow these suggestions:

- Move data set name masks to the ACS filter list.

  You can have the following VRSs:

  ```
  RMM ADDVRS DSNAME('DATA.SET.**') COUNT(31) DAYS
  RMM ADDVRS DSNAME('YCJRES.SET.**') COUNT(31) DAYS
  ```

  If so, you can assign an MC LDATE031 through ACS routines to these data set name masks, and add the following VRS:

  ```
  RMM ADDVRS DSNAME('LDATE031') COUNT(31) DAYS
  ```

- Use one VRS chain to implement one service level.

  For example, you can assign through the ACS routines the same management class (CATALOG) to all the data sets that you want to retain while they are cataloged. Then, you can add a data set name VRS for this MC:

  ```
  RMM ADDVRS DSNAME('CATALOG') WHILECATALOG
  ```

  The rest of the VRSs with the same specification can be removed.

- Code MC ACS routines to process allocations for system-managed tapes and for non-system-managed tapes.

  All the data set specifications, system-managed or non-system-managed, tape data sets are to be managed in the same way, by using the RMMVRS environment call for non-SMS tapes and regular allocations for SMS tapes.

- Assign policies by name. Move decisions out of EDGUX100.

  ACS routines are easier to change and maintain than exits, so we suggest that you include all data set specifications in the ACS routines, and remove them from the EDGUX100 exit.

  The migration from MV to MC is transparent. The MC specification overwrites the MV specification (see Figure 11-20 on page 388) and the data sets are retained by policy, for example:

  a. We create the VRS as shown here:

     ```
     RMM ADDVRS DSNAME('CYCL005') CYCLES COUNT(5)
     ```

  b. We create the data sets 1, 2, and 3, as shown in Figure 11-74, with a management value of CYCL005 assigned by EDGUX100.

  c. We add the same specification in the EDGUX100 to ACS routines, assigning to these data sets an MC=CYCL005, and we create the data sets 4, 5, and 6 in Figure 11-74. They have MC=CYCL005 and MV=CYCL005.

  d. After that, we remove the specification in EDGUX100, and we create data set 7 in Figure 11-74. This data set has only MC=CYCL005.

  e. DFSMSrmm retains five versions of the data sets (7, 6, 5, 4, and 3) although numbers 1 and 2 have only MV specification.

```
 1. Data set name YCJRES.TEST1 creation date 33 days ago with MV
 2. Data set name YCJRES.TEST1 creation date 32 days ago with MV
 3. Data set name YCJRES.TEST1 creation date 31 days ago with MV
 4. Data set name YCJRES.TEST1 creation date 29 days ago with MV and MC
 5. Data set name YCJRES.TEST1 creation date 28 days ago with MV and MC
 6. Data set name YCJRES.TEST1 creation date 25 days ago with MV and MC
 7. Data set name YCJRES.TEST1 creation date 20 days ago with MC
```

Figure 11-74 Data set examples

- Move entries in UXTABLE to ACS routines.

  The EDGUX100 exit is shipped by DFSMSrmm as SAMPLIB member EDGCVRSX. This sample exit uses a dynamic table that is initially built by the sample job EDGJSRDS. This table is called UXTABLE, and it is a list of assembler macro statements that are easily customized to add or remove entries. You can add your pooling decisions to the table by modifying the entries built by EDGCSRDS, or by adding new entries.

  If you are using UXTABLE to assign policies to data sets, you can eliminate this table by assigning an MC through ACS routines to each data set name or data set name mask with an entry in the UXTABLE.

12

DFSMSrmm automatic class selection support

Starting with Data Facility System Managed Storage removable media manager (DFSMSrmm) V2.10, you can use your storage management subsystem (SMS) automatic class selection (ACS) routines to assign a vital record specification (VRS) management class (MC) and to select a specific scratch pool. This chapter explains the steps to update your SMS environment to allow an MC to be used in place of the VRS management value (MV) assigned in the EDGUX100 user exit, and a storage group name to be used as a scratch pool name.

This chapter has the following objectives:

- Understand the interaction between DFSMSrmm and SMS
- Learn how DFSMSrmm calls SMS in both an SMS-managed environment and a non-SMS-managed environment
- Update the SMS configuration to move decisions from the EDGUX100 exit to SMS.

12.1 DFSMSrmm SMS ACS support

With DFSMSrmm, you can use your SMS ACS routines to select a specific scratch pool and storage group to be assigned to create new data on tape, and set a management class to retain and move a data set on tape. You move your policy and scratch pool decisions to your SMS ACS routines as shown in Figure 12-1. For non-system-managed tapes, DFSMSrmm calls the ACS routines to allow a management class (MC) and storage group assignment. For SMS-managed tapes, SMS calls the ACS routines directly.

To enable DFSMSrmm ACS support, you must have the SMS subsystem active and have a valid SMS configuration.

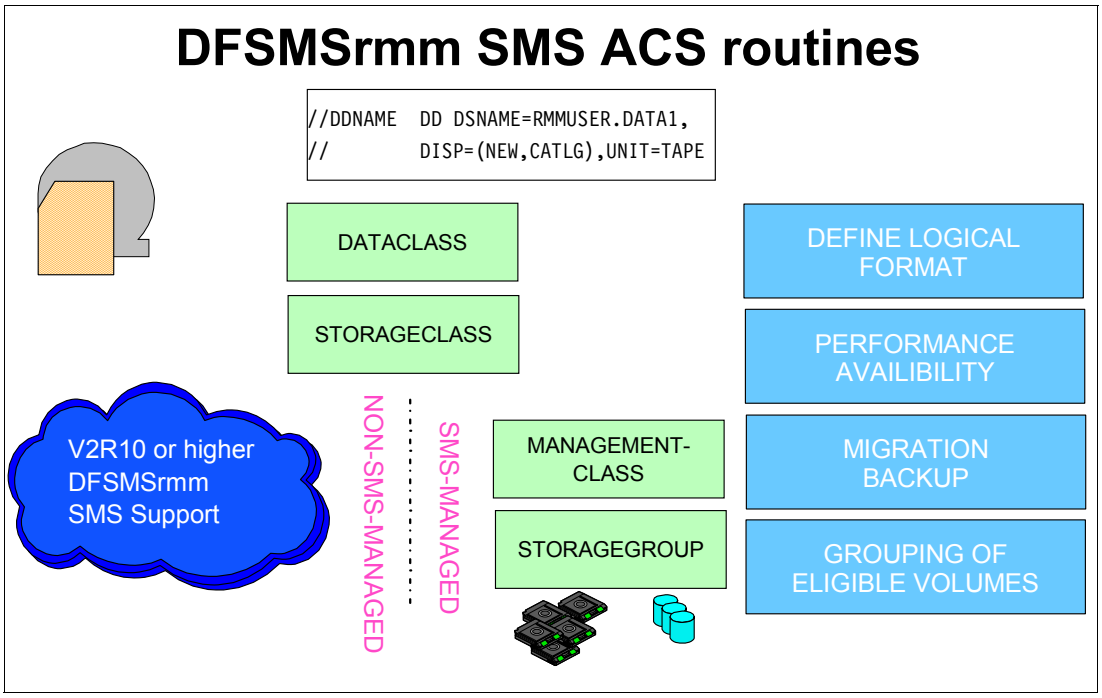


Figure 12-1 SMS ACS support with DFSMSrmm OS/390 V2R10 or later

SMS ACS routines are always called by DFSMSrmm if you have a tape mount outside an SMS-managed tape library.

ACS processing for non-system-managed tapes can be tailored by the PARMLIB member EDGRMMxx OPTION subparameters PREACS and SMSACS. Figure 12-2 shows the syntax of the EDGRMMxx OPTION subparameter.

```
                 |-PREACS(NO)---------|     |-SMSACS(NO)---------|
                 |                    |     |                    |
>>---OPTION------------------------------------------------------------>
                 |                    |     |                    |
                 |          |-NO--|    |     |          |-NO--|    |
                 |-PREACS(----------)-|     |-SMSACS(----------)-|
                            |-YES-|                     |-YES-|
```

Figure 12-2 DFSMSrmm ACS processing PARMLIB options

The PREACS and SMSACS subparameters are further explained:

- **PREACS**

  Specify the PREACS operand to control whether DFSMSrmm-supplied values and EDGUX100 installation exit-supplied values are input to SMS pre-ACS processing.

  The options are defined:

  NO Specify NO to avoid DFSMSrmm pre-ACS processing using the DFSMSrmm EDGUX100 installation exit.

  YES Specify YES to enable DFSMSrmm pre-ACS processing using the DFSMSrmm EDGUX100 installation exit.

- **SMSACS**

  The scratch pooling and policy assignment rules are by default coded in the EDGUX100 user exit. You can move your decisions to the SMS ACS routines where you can use ACS input variables as a base for the MC and storage group assignment.

  The options are defined:

  NO Specify NO to prevent DFSMSrmm from calling the SMS ACS processing to obtain MC and storage group names. DFSMSrmm system-based scratch pooling and VRS management values based on the EDGUX100 installation exit are used.

  YES Specify YES to enable DFSMSrmm calls to the SMS ACS processing to obtain MC and storage group names. If values are returned by the SMS ACS routines, the values are used instead of the DFSMSrmm and EDGUX100 decisions.

  We suggest that you move pooling decisions and VRS management value assignments out of the EDGUX100 user exit.

Suggestion: Do not implement PREACS processing. Also, do not use the EDGUX100 user exit, except for the following reasons:

- Cartridge loader control (access control list (ACL))
- Data set recording
- Sticky label customization
- Request to ignore volumes

12.1.1 How ACS support works

Use ACS routines for scratch pooling based on tape storage group names. Using ACS processing to set a storage group name overrides all other pool selection methods.

DFSMSrmm provides support for non-system-managed tape and for system-managed manual tape libraries. This support enables pooling at the individual volume level. You assign a storage group name to each volume by using DFSMSrmm TSO subcommands. Or, you can assign a storage group name to each volume by using pooling information that you define with the DFSMSrmm EDGRMMxx PARMLIB VLPOOL command (see “Tailoring your EDGRMMxx PARMLIB member” on page 464).

DFSMSrmm calls ACS routines to pass environment information, including the pool identified by DFSMSrmm system-based pooling. The ACS routine can optionally set a storage group name, which overrides the DFSMSrmm system-based pool.

12.1.2 ACS support for non-system-managed volumes

In a non-system-managed library, DFSMSrmm supports SMS ACS routines in different ways:

- DFSMSrmm directly calls the ACS routines if you specified SMSACS(YES) in the EDGRMMxx PARMLIB member to allow an MC and a storage group to be assigned. Figure 12-3 shows how it works.

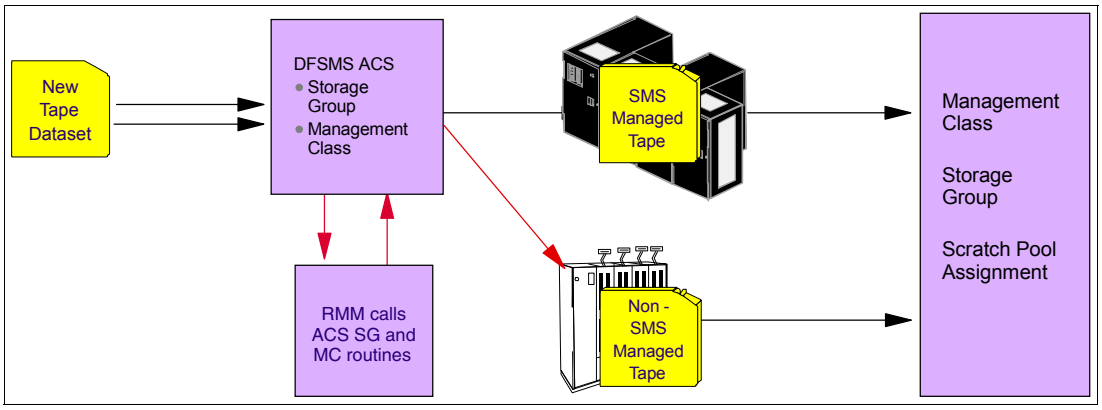


Figure 12-3 How DFSMSrmm SMS direct calls for non-system-managed volumes

- You can use the EDGUX100 user exit to call SMS ACS routines to assign a VRS management class value and a scratch pool if you have specified PREACS(YES) as shown in Figure 12-4.

  Use pre-ACS processing to obtain the DFSMSrmm system-based pool or the EDGUX100 installation exit pool prefix as an input value to the ACS routines in the MSPOOL read-only variable. During pre-ACS processing, DFSMSrmm does not make the RMMPOOL environment call to the ACS routine. During pre-ACS processing for new allocations, DFSMSrmm performs the following actions:

  – Uses the VLPOOL definitions to select a default DFSMSrmm pool using DFSMSrmm system-based pooling. DFSMSrmm sets a pool prefix if a specific pool is selected.

  – Calls the EDGUX100 installation exit to obtain a pool prefix value. If a pool prefix value is returned, the pool prefix value that is returned by the EDGUX100 installation exit overrides the DFSMSrmm selected pool.

  – Returns the selected value in the MSPOOL read-only variable if the MSPOOL variable is not already set by the pre-ACS exit.

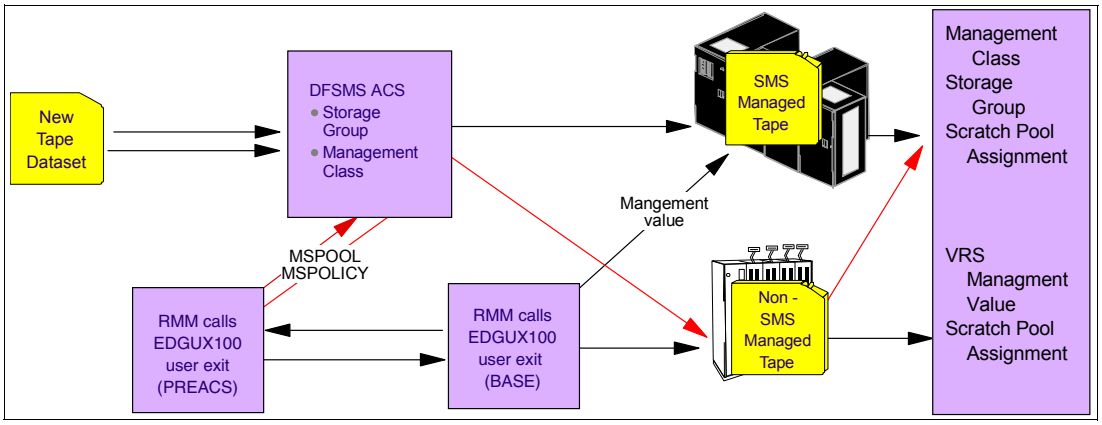


Figure 12-4 DFSMSrmm EDGUX100 user exit call with PREACS(YES)

- You can use both the DFSMSrmm direct call and the user exit EDGUX100 call together to take the decision as shown in Figure 12-5.

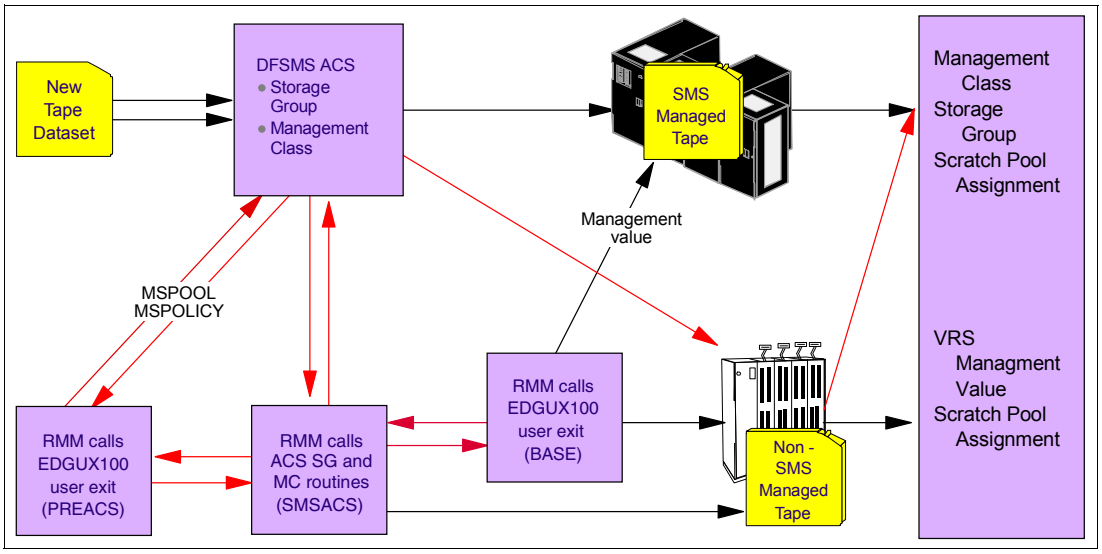


Figure 12-5 DFSMSrmm SMS direct and EDGUX100 user exit call

Note: To enable DFSMSrmm ACS support, you must have the SMS subsystem active and have a valid SMS configuration.

If you are using PREACS processing and have installed the EDGUX100 user exit, DFSMSrmm processes the functions in the following sequence:

1. ACS processing calls the PRE-ACS exit.
2. DFSMSrmm calls EDGUX100 and sets MSPOOL and MSPOLICY.
3. ACS decides whether the entity is system managed. During processing, the four PRE-ACS read-only variables can be used as input to the decision.
4. For non-system-managed tape, we enter DFSMSrmm ACS support. ACS processing is called for storage group processing, and again for management class processing.
5. DFSMSrmm calls EDGUX100 for its basic exit function.

12.1.3 Non-system-managed tape libraries

A non-system-managed tape library consists of all the volumes, shelves, and drives that are not in an Automated Tape Library Dataserver or manual tape library. Think of a non-system-managed tape library as the traditional tape library in a data center, or as an automated environment that is not system-managed. DFSMSrmm provides complete tape management functions for the volumes and shelves in this traditional tape library.

All tape media and drives supported by z/OS are supported in this environment. Use DFSMSrmm to fully manage all types of tapes in a non-system-managed tape library, including 3420 reels, 3480, 3490, 3590, and 3592 cartridge system tapes.

You can also use DFSMSrmm to manage volumes in any automated tape library that has special software, including an IBM Tape Library Dataserver that is managed using Basic Tape Library Support (BTLS) or a VTF Mainframe (VTFM).

Most non-IBM libraries do not provide support for system-managed tape, so we consider them to be the traditional tape libraries.

The intended IBM direction is to replace the DFSMSrmm exit function with an alternative, either the SMS policy or parmlib option.

12.1.4 SMS read-only variables

DFSMSrmm calls the ACS routines to request the assignment of storage group and management class names for non-system-managed tape data sets. Table 12-1 lists the read-only variables that are set for DFSMSrmm requests.

Table 12-1 SMS read-only variables set by DFSMSrmm

Variables			
&ACCT_JOB	&ACCT_STEP	&ACSENVIR	&DD
&DSN	&DSORG	&DSTYPE	&EXPDT
&FILENUM	&GROUP	&HLQ	&JOB
&LABEL	&LIBNAME	&LLQ	&NQUAL
&PGM	&STORGRP	&SYSNAME	&SYSPLEX
&UNIT	&USER	&XMODE	

Note: The SMS pre-ACS exit &MSPOOL, &MSPOLICY, &MSDEST, and &MSPARM read-only variables can be used by a tape management system only.

When the &ACSENVIR variable is set to RMMPOOL, DFSMSrmm requests SMS to return a storage group name. DFSMSrmm requests that both the management class and storage group ACS routines are run. With a combination of these routines, you can decide whether you want SMS to return a storage group value, and what the value will be. If a storage group name is returned, it must be a valid tape storage group name. Using different &ACSENVIR values helps you differentiate between DFSMSrmm requests for a storage group name and allocation requests for system-managed data sets.

When the &ACSENVIR variable is set to RMMVRS, DFSMSrmm requests SMS to return a management class name. DFSMSrmm requests that only the management class ACS routine is run. Using different &ACSENVIR values helps you differentiate between DFSMSrmm requests for a management class name and allocation requests for system-managed data sets.

Using the read-only variables

Most ACS variables are read-only. Read-only variables contain data set and system information, and they reflect what is known at the time of the allocation request. You can use read-only variables in comparison operations, but you cannot change their values.

Note: The read-only variables are case-sensitive.

The following variables are read-only variables:

- **&ACCT_JOB**

  The accounting information from the JOB statement.

  Type: Literal

  Maximum value: 142 characters

- **&ACCT_STEP**

  The accounting information.

  Type: Literal

  Maximum value: 142 characters

- **&ACSENVIR**

  The environment on which the ACS routine was invoked (one of the following environments):

RECALL	For data set recall operations
RECOVER	For data set recovery operations
RENAME	For data set “alter rename” operations
RMMPOOL	For DFSMSrmm requests for a storage group name
RMMVRS	For DFSMSrmm requests for a management class name
CONVERT	For data set “convert in place” operations
ALLOC	For new data set allocations (this is the default)
STORE	OSREQ object store environment
CHANGE	OSREQ object change environment
CTRANS	OSMC object class transition environment
Other	Installation exit can set its own value before re-invoking ACS

  Type: Literal

  Maximum value: Eight characters

- **&DD**

  DDNAME in the DD statement of the data set.

  Type: Literal

  Maximum value: Eight characters

- **&DSN**

  The name of the data set or collection for which ACS processing is taking place. For Virtual Storage Access Method (VSAM) data sets, only the cluster name is passed to the ACS routine; the component names are not.

  If the data set has an absolute or relative generation number, it is stripped from &DSN. The generation number is the low-level qualifier of the data set name.

  Type: Literal

  Maximum value: 44 characters

- **&DSORG**

  The data set organization (one of these options):

PS	Physical sequential
PO	Partitioned
VS	VSAM organization
DA	Basic direct-access method (BDAM) organization
Null	No value specified

  Type: Literal

Maximum value: Two characters

- **&DSTYPE**

  The data set type (one of the following types):

GDS	One generation data set of a generation data group, or any data set allocated with a relative generation number (such as A.B.C(+1)) or an absolute generation number (such as A.B.C.G0000V00).
PERM	Standard permanent data sets.
TEMP	Temporary data sets.
Null	None of the above.

  Type: Literal

  Maximum value: Eight characters

- **&EXPDT**

  The expiration date in the form of *YYYYDDD* where *YYYY* is the four-digit number for the year. The maximum allowable value for *yyyy* is 9799. DDD is the three-digit number for the day of the year from 1 - 366.

  > **Exception:** Expiration dates of 99365 and 99366 are considered “NEVER-SCRATCH” dates.

  Type: Literal

  Maximum value: Seven characters

- **&FILENUM**

  The value of the FILENUM ACS read-only variable. This variable corresponds to the data set sequence number on the JCL LABEL parameter. The default is 1. This field is optional.

  Type: Numeric

  Maximum value: Four characters

- **&GROUP**

  The RACF-defined group to which you are connected, or the group specified in the GROUP keyword on the JCL JOB statement. If the environment is “recall” or “recover”, &GROUP is set only if the requester of the recall or recover is not a DFSMShsm authorized user. When DFSMShsm invokes the ACS routines, &GROUP is the group associated with &USER.

  Type: Literal

  Maximum value: Eight characters

- **&HLQ**

  The high-level (first) qualifier of the data set or collection name.

  Type: Literal

  Maximum value: Eight characters

- **&JOB**

  The job name, the started task name, or the TSO/E user ID from the JOB statement, depending on the execution mode (&XMODE).

  Type: Literal

  Maximum value: Eight characters

- **&LABEL**

  The value of the LABEL ACS read-only variable. This variable corresponds to the label field of the JCL LABEL parameter. Allowable values are NL, AL, SL, NSL, SUL, AUL, BLP, LTM, or blank. The default is IBM Standard Label. This field is optional.

  Type: Literal

  Maximum value: Three characters

- **&LIBNAME**

  The name for the LIBNAME ACS read-only variable. It can contain a 1 - 8 character tape library name. This field is optional.

  Type: Literal

  Maximum value: Eight characters

- **&LLQ**

  The low-level (last) qualifier of the data set or collection name.

  Type: Literal

  Maximum value: Eight characters

- **&MSPDEST**

  The destination, specified in data set name format, for a tape management system-driven tape allocation. This value is specified through the access method services (AMS) pre-ACS installation exit. The data set name format lets you specify a sequence of destinations to be identified, where each qualifier is a specific destination. For example, a data set vaulted first at location OUTD and then sent to OLTS can have an MSPDEST of 'OUTD.OLTS'. The actual values depend on the support provided by your tape management system.

  Type: Alphanumeric

  Maximum value: 44 characters

- **&MSPARM**

  Additional information that is related to a tape management system-driven tape allocation. This is a variable length field that can be indexed. The value is specified through an external exit.

  Type: Alphanumeric

  Maximum value: 256 bytes, including a 2-byte length field for each value specified

- **&MSPOLICY**

  The name of a management policy associated with tape data for a tape management system-driven allocation. You can use the DFSMSrmm EDGUX100 installation exit to set MSPOLICY to a VRS management value name. You can also set the value of this variable using the SMS pre-ACS installation exit, or allow your tape management system to set it using the pre-ACS installation exit.

  Type: Alphanumeric

  Maximum value: Eight characters

- **&MSPOOL**

  A tape pool name associated with the data set being allocated. In a system-managed tape environment with scratch pool support, you can use this variable to specify a default storage group, where the tape storage group is equivalent to the tape pool specified in the variable. If you use the DFSMSrmm EDGUX100 installation exit, you can set this variable

to the pool name or prefix that is determined by the DFSMSrmm scratch pool processing. This variable can also be set through the pre-ACS installation exit.

Type: Alphanumeric

Maximum value: Eight characters

- **&NQUAL**

  The number of qualifiers in the data set or collection name.

  Type: Numeric

  Maximum value: 22

- **&PGM**

  The name of the program that the system is running.

  Type: Literal

  Maximum value: Eight characters

- **&SYSNAME**

  Specifies the system name of the system on which the ACS routine is executing. This field is optional.

  Type: Literal

  Maximum value: Eight characters

- **&SYSPLEX**

  Specifies the IBM Parallel Sysplex® name of the system on which the ACS routine is executing. This field is optional.

  Type: Literal

  Maximum value: Eight characters

- **&UNIT**

  IBM-supplied or installation-defined generic name for a device type (for example, 3380, SYSDA).

  Type: Literal

  Maximum value: Eight characters

  A slash (/) preceding a four-digit number represents a unit address (for example, /3380). When you allocate a tape data set with DISP=MOD, and no unit information is specified in the JCL, this variable is blank and SMS might attempt to manage the tape data set as a DASD-resident data set.

- **&USER**

  The user ID of the person allocating the data set. When DFSMShsm invokes the ACS routines, &USER is either the requester of the recall or recover, or the user ID of the DFSMShsm address space. If the environment is “recall” or “recover”, the variable is set only if the requester of the recall or recover is not a DFSMShsm authorized user.

  Type: Literal

  Maximum value: Eight characters

- **&XMODE**

  The execution mode in which the data set is being allocated (one of the following modes):

BATCH	Batch execution mode
TSO	TSO execution mode
TASK	A started address space

Type: Literal

Maximum value: Eight characters

12.1.5 Implementing SMS ACS processing

You can move your scratch pool and policy assignment decisions to the SMS ACS routines, where you can use ACS input variables as a base for management class and storage group assignments. In the EDGRMMxx PARMLIB member, specify **SMSACS(YES)** so that DFSMSrmm calls the SMS ACS routines for pooling decisions and policy assignment.

Before you enable SMS ACS support in your installation, you must make some changes in your SMS ACS routines.

Defining your SMS environment

To define an SMS environment for non-system-managed tapes by using the ISMF dialog as shown in Figure 12-6, use the steps in the following sections.

```
                    ISMF PRIMARY OPTION MENU - z/OS DFSMS V1 R12
Selection or Command ===>

0  ISMF Profile              - Specify ISMF User Profile
1  Data Set                  - Perform Functions Against Data Sets
2  Volume                    - Perform Functions Against Volumes
3  Management Class          - Specify Data Set Backup and Migration Criteria
4  Data Class                - Specify Data Set Allocation Parameters
5  Storage Class             - Specify Data Set Performance and Availability
6  Storage Group             - Specify Volume Names and Free Space Thresholds
7  Automatic Class Selection - Specify ACS Routines and Test Criteria
8  Control Data Set          - Specify System Names and Default Criteria
9  Aggregate Group           - Specify Data Set Recovery Parameters
10 Library Management        - Specify Library and Drive Configurations
11 Enhanced ACS Management   - Perform Enhanced Test/Configuration Management
C  Data Collection           - Process Data Collection Function
G  Report Generation         - Create Storage Management Reports
L  List                      - Perform Functions Against Saved ISMF Lists
P  Copy Pool                 - Specify Pool Storage Groups for Copies
R  Removable Media Manager   - Perform Functions Against Removable Media
X  Exit                      - Terminate ISMF
Use HELP Command for Help; Use END Command or X to Exit.
```

Figure 12-6 ISMF primary option menu

Defining a tape library

If you do not have a tape library, or do not want to associate the SMS storage groups with an existing library, you must define a new library to ISMF. You can do this using the ISMF define library application, but you specify a made-up library ID. For other data fields required for tape library definition, you can select any values that you want. When you do this, OAM issues a CBR3006I message at startup time.

Note: You can skip this section if you do not make the storage group assignments.

Follow these steps to define a library:

1. Select Option 10 Library Management by entering 10 on the Selection or Command line on the ISMF PRIMARY OPTION MENU panel (see Figure 12-6) to display the LIBRARY MANAGEMENT SELECTION MENU.

2. Select Option 3 Tape Library by entering `3` on the Selection or Command line as shown in Figure 12-7 to display the TAPE LIBRARY APPLICATION SELECTION panel.

```
                        LIBRARY MANAGEMENT SELECTION MENU
 Enter Selection or Command ===>

 1  Optical Library         - Optical Library Configuration
 2  Optical Drive           - Optical Drive Configuration
 3  Tape Library            - Tape Library Configuration

 Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-7 ISMF library selection menu

Define a system-managed tape library with a made-up library ID using the TAPE LIBRARY APPLICATION SELECTION panel as shown in Figure 12-8.

```
                       TAPE LIBRARY APPLICATION SELECTION
 Command ===>

 To Perform Library operations, Specify:

   CDS Name  . . . . . . . 'SYS1.SMS.SCDS'
                              (1 to 44 Character Data Set Name or 'Active')
   Library Name  . . . . . $VTFM          (For Tape Library list, fully or
                                          Partially Specified or * for all)

 Select one of the following options  :
   3  1. List    - Generate a list of Libraries
      2. Display - Display a Library
      3. Define  - Define a Library
      4. Alter   - Alter a Library

 If List option is chosen,
    Enter "/" to select option      Respecify View Criteria
                                    Respecify Sort Criteria

 Use ENTER to Perform Selection;
 Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-8 Specifying a library name to SMS for a non-system-managed library

Note: In the Library Name field, enter the SMS-compatible name for your tape library. This name relates your tape library to your SMS tape storage group, which you define later. There is a minor restriction when you name the library. The first character of a library name must not be `V` or one of the DFSMSrmm-defined locations (LOCAL, REMOTE, or DISTANT).

3. In the TAPE LIBRARY DEFINE panel (see Figure 12-9 on page 453), specify a Library ID, the Entry Default Use Attribute, and the Eject Defaults. Press Enter to perform verification. Type `END` to save and exit the dialog.

   The Description is a 120-byte field that allows you to enter a description of the library definition for use by the installation. There are no restrictions on its content. The use of the Description field is optional.

```
                              TAPE LIBRARY DEFINE                 Page 1 of 4
Command ===>

SCDS Name  . : SYS1.SMS.SCDS
Library Name : $VTFM

To Define Library, Specify:

  Description ==>
              ==>
  Library ID . . . . . . . . . . . FFFFF     (00001 to FFFFF)
  Console Name . . . . . . . . . .
  Entry Default Data Class . . . .
  Entry Default Use Attribute  . . P         (P=PRIVATE or S=SCRATCH)
  Eject Default  . . . . . . . . . P         (P=PURGE or K=KEEP)

Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-9 Defining a new SMS tape library (part 1 of 4)

Note: You can use any library ID that is not used in your installation.

4. In the second panel, you can specify a scratch threshold for each media type as shown in Figure 12-10.

```
                              TAPE LIBRARY DEFINE                 Page 2 of 4
Command ===>

SCDS Name  . : SYS1.SMS.SCDS
Library Name : $VTFM

Media Type:                 Scratch Threshold
  Media1: . . . . . . . . . . .      0   (0 to 999999)
  Media2: . . . . . . . . . . .      0   (0 to 999999)
  Media3: . . . . . . . . . . .      0   (0 to 999999)
  Media4: . . . . . . . . . . .      0   (0 to 999999)
  Media5: . . . . . . . . . . .      0   (0 to 999999)
  Media6: . . . . . . . . . . .      0   (0 to 999999)
  Media7: . . . . . . . . . . .      0   (0 to 999999)
  Media8: . . . . . . . . . . .      0   (0 to 999999)
  Media9: . . . . . . . . . . .      0   (0 to 999999)
  Media10:  . . . . . . . . . .      0   (0 to 999999)
  Media11:  . . . . . . . . . .      0   (0 to 999999)
  Media12:  . . . . . . . . . .      0   (0 to 999999)
  Media13:  . . . . . . . . . .      0   (0 to 999999)

Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-10 Defining a new SMS tape library (part 2 of 4)

5. Check your initial online status settings if you have more than one system sharing this SMS environment. To do so, scroll down the current panel that is displayed and set the Initial Online Status for all displayed systems to `YES` and press Enter. See Figure 12-11 on page 454.

```
                                TAPE LIBRARY DEFINE                 Page 3 of 4
Command ===> Y

SCDS Name  . : SYS1.SMS.SCDS
Library Name : $VTFM

  Initial Online Status (Yes, No, or Blank):
    SC63     ===>      SC64     ===>      SC65     ===>      SC70     ===> YES

Warning:
  When you connect a tape library to a system group rather than a system,
  you lose the ability to vary that library online or offline to the
  individual systems in the system group. It is strongly recommended that
  the tape library be connected to individual systems only.

Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-11 Defining a new SMS tape library (part 3 of 4)

6. Restart your object access method (OAM) dynamically by using the MVS MODIFY OAM, RESTART command that is shown in Figure 12-12.

```
MODIFY OAM,RESTART
```

Figure 12-12 Restart OAM address space (part 4 of 4)

In Figure 12-12, OAM is the name of the OAM started task. Figure 12-13 shows the messages you receive after you successfully restart OAM.

```
CBR3006I Library $VTFM with Library ID FFFFF unknown in I/O
configuration.
CBR3002E Library $VTFM no longer usable.
CBR0097I OAM restart completed.
```

Figure 12-13 OAM restart messages

7. Use the MVS DISPLAY SMS command shown in Figure 12-14 to see the newly defined library.

```
D SMS,LIB(ALL),DETAIL
```

Figure 12-14 Display the SMS-managed tape libraries status

Figure 12-15 shows the result of the MVS DISPLAY SMS command.

```
CBR1110I OAM library status: 389
TAPE      LIB  DEVICE    TOT  ONL  AVL  TOTAL  EMPTY SCRTCH  ON OP
LIBRARY   TYP  TYPE      DRV  DRV  DRV  SLOTS  SLOTS   VOLS
LIB1      AL    349-4L1    4    0    0   1445     32      6  Y  Y
§VTFM     UNK              0    0    0      0      0      0  N  N
```

Figure 12-15 Output of a DISPLAY SMS command

Note: Deleting a tape library by using ISMF panels has no effect on the tape configuration database (TCDB). Instead, the library definition is removed only from the specified source control data set (SCDS). To delete a tape library from the TCDB, use the IDCAMS DELETE command.

Defining a tape storage group

To use SMS storage groups, first define tape storage groups by using ISMF. To define tape storage groups, you need at least one system-managed tape library defined to ISMF. See "Defining a tape storage group" on page 455.

The storage group type TAPE is provided to classify tape cartridges in DFSMS. A *tape storage group* consists of tape libraries and the tape cartridges that are associated with them. A tape storage group can contain 1- 8 tape libraries specified by their library name, and one tape library can contain more than one tape storage group.

Note: You can skip this section if you do not make storage group assignments.

To define a storage group, follow these steps:

1. Select Option 6 Storage Group by entering `6` on the Selection or Command line on the ISMF PRIMARY OPTION MENU, as shown in Figure 12-6 on page 451, to display the STORAGE GROUP APPLICATION SELECTION menu.
2. On the next panel, which is shown in Figure 12-16, specify the Storage Group Name and Storage Group Type. The Storage Group Type is `TAPE` in our case.
3. Select Option 2 Define by entering `2` on the Command line to display the TAPE STORAGE GROUP DEFINE panel.

```
                     STORAGE GROUP APPLICATION SELECTION
Command ===>

To perform Storage Group Operations, Specify:
  CDS Name  . . . . . . 'SYS1.SMS.SCDS'
                                (1 to 44 character data set name or 'Active' )
  Storage Group Name    VTFMSG            (For Storage Group List, fully or
                                          partially specified or * for all)
  Storage Group Type    TAPE              (VIO, POOL, DUMMY, COPY POOL BACKUP,
                                          OBJECT, OBJECT BACKUP, or TAPE)

Select one of the following options  :
  2  1. List    - Generate a list of Storage Groups
     2. Define  - Define a Storage Group
     3. Alter   - Alter a Storage Group
     4. Volume  - Display, Define, Alter or Delete Volume Information

If List Option is chosen,
   Enter "/" to select option      Respecify View Criteria
                                   Respecify Sort Criteria
Use ENTER to Perform Selection;
Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-16 Specifying a storage group name to SMS

4. Figure 12-17 shows the TAPE STORAGE GROUP DEFINE panel. The SCDS Name and Storage Group Name fields are output fields containing the SCDS and storage group names, which you specified in the Storage Group Application Selection panel. The Description field is an optional field of 120 characters in which you can describe the tape storage group. ISMF primes the field with blanks, which are the default.

   The Library Names field is used to specify the tape libraries that own the volumes within this storage group. One to eight library names can be associated with a tape storage group. At least one library name must be specified when defining a tape storage group. The library name in the tape storage group definition must also be defined in the same SCDS.

   Press Enter to perform verification.

   Type the `END` command to save the newly defined tape storage group and return to the STORAGE GROUP APPLICATION SELECTION panel.

```
                          TAPE STORAGE GROUP DEFINE
 Command ===>

 SCDS Name . . . . . : SYS1.SMS.SCDS
 Storage Group Name  : VTFMSG

 To DEFINE Storage Group, Specify:

   Description ==> FOR THE IBM VIRTUAL TAPE FACILITY FOR MAINFRAME (VTFM)
               ==>

   Library Names  (1 to 8 characters each):
     ===> $VTFM     ===>           ===>           ===>
     ===>           ===>           ===>           ===>

   DEFINE    SMS Storage Group Status . . Y  (Y or N)

 Use ENTER to Perform Verification and Selection;
 Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-17 Defining a new SMS storage group

5. Repeat the preceding steps 1 - 4 for each storage group that you have to define.

   Note: Each unique DFSMSrmm volume pool is selected using SMS ACS routines. You have to define an SMS storage group for each DFSMSrmm volume pool.

6. Check your SMS storage group status settings if you have more than one system sharing this SMS environment. To do so, in the DEFINE SMS Storage Group Status field in the panel shown in Figure 12-18 on page 457, specify `Y` and then press Enter. The settings for each system need to match the settings of your SMS tape libraries.

```
                        SMS STORAGE GROUP STATUS ALTER         Page 1 of 2
Command ===>

SCDS Name  . . . . : SYS1.SMS.SCDS
Storage Group Name : VTFMSG
Storage Group Type : TAPE
To ALTER Storage Group System/                    ( Possible SMS SG Status
Sys Group Status, Specify:                          for each:
                                                    - Pool SG Type
 System/Sys      SMS SG   System/Sys      SMS SG       NOTCON, ENABLE, DISALL
 Group Name      Status   Group Name      Status       DISNEW, QUIALL, QUINEW
 ----------      ------   ----------      ------    - Tape SG Type
 SC63       ===> NOTCON   SC64       ===> NOTCON       NOTCON, ENABLE,
 SC65       ===> NOTCON   SC70       ===> ENABLE       DISALL, DISNEW
            ===>                     ===>          - Copy Pool Backup SG Type
            ===>                     ===>              NOTCON, ENABLE )
            ===>                     ===>          * SYS GROUP = sysplex
            ===>                     ===>            minus Systems in the
            ===>                     ===>            Sysplex explicitly
            ===>                     ===>            defined in the SCDS
Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-18 SMS STORAGE GROUP STATUS DEFINE panel

If you successfully defined all your storage groups, you can display them by selecting Option 1 Generate a List Storage Group on the STORAGE GROUP APPLICATION SELECTION menu (see Figure 12-16 on page 455). Specify an asterisk (*) in the STORAGE GROUP NAME field to show all defined storage groups in your SMS environment. Figure 12-19 shows the result of the list.

```
                                STORAGE GROUP LIST
Command ===>                                                Scroll ===> HALF
                                                         Entries 1-1 of 1
                                                         Data Columns 3-6 of 48
CDS Name : SYS1.SMS.SCDS

Enter Line Operators below:

     LINE       STORGRP  SG                VIO      VIO   AUTO
     OPERATOR   NAME     TYPE              MAXSIZE  UNIT  MIGRATE
    ---(1)----  --(2)--- -------(3)------  --(4)--  (5)-  --(6)---
                VTFMSG   TAPE              -------  ----  --------
    ----------  -------- ------  BOTTOM  OF  DATA  ------  --------  ----------
```

Figure 12-19 Listing all defined SMS storage groups

Before you activate the new settings, validate them by using the ISPF dialog. Select Option 8 Control Data Set by entering 8 on the Selection or Command line on the ISMF PRIMARY OPTION MENU panel (see Figure 12-6 on page 451) to see the CDS APPLICATION SELECTION panel shown in Figure 12-20 on page 458. Then, select Option 4 Validate by entering 4 on the Command line.

```
                          CDS APPLICATION SELECTION       ACTIVATION SCHEDULED
Command ===>

To Perform Control Data Set Operations, Specify:
  CDS Name . . 'SYS1.SMS.SCDS'
                            (1 to 44 Character Data Set Name or 'Active')

Select one of the following Options:
  4  1. Display       - Display the Base Configuration
     2. Define        - Define the Base Configuration
     3. Alter         - Alter the Base Configuration
     4. Validate      - Validate the SCDS
     5. Activate      - Activate the CDS
     6. Cache Display - Display CF Cache Structure Names for all CF Cache Sets
     7. Cache Update  - Define/Alter/Delete CF Cache Sets
     8. Lock Display  - Display CF Lock Structure Names for all CF Lock Sets
     9. Lock Update   - Define/Alter/Delete CF Lock Sets
If CACHE Display is chosen, Enter CF Cache Set Name . . *
If LOCK Display is chosen, Enter CF Lock Set Name . . . *
                             (1 to 8 character CF cache set name or * for all)
Use ENTER to Perform Selection;
Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-20 Validate the SCDS definitions

Figure 12-21 shows how you can specify a listing of the data set to get all the messages. Press Enter.

```
                       VALIDATE ACS ROUTINES OR ENTIRE SCDS
Command ===>

To Perform Validation, Specify:

  SCDS Name  . . . . . 'SYS1.SMS.SCDS'
                           (1 to 44 Character Data Set Name)

  ACS Routine Type . . *   (DC=Data Class, MC=Management Class, SC=Storage
                           Class, SG=Storage Group, *=Entire SCDS)

  Listing Data Set . . LISTING
                           (1 to 44 Character Data Set Name)
Use ENTER to Perform Validation;
Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-21 Process SCDS validation

If the validation is finished, the result is displayed as shown in Figure 12-22 on page 459.

```
 BROWSE    MHLRES7.LISTING                          Line 00000000 Col 001 080
 Command ===>                                                Scroll ===> PAGE
********************************* Top of Data **********************************
                              VALIDATION RESULTS

VALIDATION RESULT:  VALIDATION SUCCESSFUL - WARNINGS DETECTED
SCDS NAME:          SYS1.SMS.SCDS
ACS ROUTINE TYPE:   *
DATE OF VALIDATION: 2010/10/17
TIME OF VALIDATION: 16:58

IGD06023I STORAGE GROUP MHLEAV IS NOT REFERENCED BY THE STORAGE GROUP ACS ROUTINE
IGD06023I STORAGE GROUP SGMHL02 IS NOT REFERENCED BY THE STORAGE GROUP ACS ROUTINE
IGD06023I STORAGE GROUP SGMHL04 IS NOT REFERENCED BY THE STORAGE GROUP ACS ROUTINE
IGD06023I STORAGE GROUP TPCDB2 IS NOT REFERENCED BY THE STORAGE GROUP ACS ROUTINE
```

Figure 12-22 Validation result messages

After all errors are fixed, go back to the CDS APPLICATION SELECTION panel (see Figure 12-20 on page 458).

You activate the new SMS configuration dynamically by using the MVS SETSMS command as shown in Figure 12-23.

```
SETSMS SCDS(SYS1.SMS.SCDS)
```

Figure 12-23 Activate your new SMS settings

In Figure 12-23, SCDS is your currently used SMS source data set. You see the IDG008I message as shown in Figure 12-24 after the successful activation.

```
RESPONSE=SC65      IGD008I NEW CONFIGURATION ACTIVATED FROM SCDS
RESPONSE=SYS1.SMS.SCD
```

Figure 12-24 Activate SMS setting messages

Use the DISPLAY SMS command as shown in Figure 12-25 to display the newly defined manual tape library.

```
D SMS,LIB(ALL),DETAIL
```

Figure 12-25 Display SMS library command

Figure 12-26 shows the result of the Display SMS command.

```
CBR1110I OAM library status: 116
TAPE      LIB  DEVICE    TOT  ONL  AVL   TOTAL   EMPTY SCRTCH  ON OP
LIBRARY   TYP  TYPE      DRV  DRV  DRV   SLOTS   SLOTS   VOLS
$VTFM     UNK              0    0    0       0       0      0  Y  N
LIB1      UNK              0    0    0    1445    1427      0  Y  N
LIB2      AL   3584-L22   16    0    0     260      84     22  Y  Y
```

Figure 12-26 Output of the Display SMS library command

Defining management classes

Because no management class attributes are available for tape cartridges, defining management classes is optional. You have to define policies for management classes by defining VRSs. Use the RMM subcommand ADDVRS with the DSNAME operand, or the DFSMSrmm Add Data Set VRS panel in the DFSMSrmm ISPF dialog to define VRSs. Use the data set name masks that match the management class names you have defined.

Note: You can skip this section if you do not make management class assignments.

In our examples, we define a management class to move the special meaning expiration date decision from the EDGUX100 user exit to SMS ACS. We define a management class of M99000 (catalog control). You can change the first qualifier from “M” to “D” if you migrate from EDGUX100 to SMS ACS and if you previously defined all the VRSs to match the special meaning expiration dates starting with a “D”.

Note: There are no differences in the VRS definitions to match to a management class or to match a VRS management value.

To define a management class, the following steps are required:

1. Select Option 3 Management Class by entering 3 on the Selection or Command line on the ISMF PRIMARY OPTION MENU panel as shown in Figure 12-6 on page 451 to display the MANAGEMENT CLASS APPLICATION SELECTION panel.
2. To define a management class, you must specify a CDS Name and a Management Class Name on the panel shown in Figure 12-27. The CDS Name must be the name of an SCDS. ISMF primes the field with the last used name. The default is the quoted word 'ACTIVE', which represents the currently active configuration, but you cannot define or alter management classes to the 'ACTIVE' configuration. In the Management Class Name field, you must specify the name of the management class that you are defining.

```
                    MANAGEMENT CLASS APPLICATION SELECTION        Page 1 of 2
 Command ===>

 To perform Management Class Operations, Specify:
   CDS Name  . . . . . . . . . 'SYS1.SMS.SCDS'
                               (1 to 44 character data set name or 'Active' )
   Management Class Name . . . M99000    (For Management Class List, fully or
                                         partially specified or * for all)

 Select one of the following options  :
   3  1. List    - Generate a list of Management Classes
      2. Display - Display a Management Class
      3. Define  - Define a Management Class
      4. Alter   - Alter a Management Class

 If List Option is chosen,
    Enter "/" to select option      Respecify View Criteria
                                    Respecify Sort Criteria


 Use ENTER to Perform Selection; Use DOWN Command to View next Selection Panel;
 Use HELP Command for Help; Use END Command to Exit.
```

Figure 12-27 ISMF management class selection

3. Select Option 3 DEFINE by entering 3 on the Command line on the panel to display the MANAGEMENT CLASS DEFINE panel <<wrong panel in figure 12-28 > that is shown in Figure 12-28.

```
                          MANAGEMENT CLASS ALTER              Page 1 of 5
 Command ===>

 SCDS Name . . . . . . : SYS1.SMS.SCDS
 Management Class Name : M99000

 To ALTER Management Class, Specify:

   Description ==> Special meaning expdt 99000 whilecatalog
               ==> for DFSMSrmm only

   Expiration Attributes
     Expire after Days Non-usage  . . NOLIMIT     (1 to 9999 or NOLIMIT)
     Expire after Date/Days . . . . . NOLIMIT     (0 to 9999, yyyy/mm/dd or
                                                  NOLIMIT)

   Retention Limit  . . . . . . . . . NOLIMIT     (0 to 9999 or NOLIMIT)



 Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
 Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 12-28 ISMF panel: MANAGEMENT CLASS DEFINE

You can use the optional description field to describe the management class. All other fields are not used for a management class defined for DFSMSrmm use.

Tip: You do not have to define any parameters in the ISMF definition panels because the management class attributes are not used by DFSMSrmm.

You can restart your SMS system dynamically by using the MVS SETSMS command shown in Figure 12-29.

```
SETSMS SCDS(SYS1.SMS.SCDS)
```

Figure 12-29 Activate your new SMS settings

In Figure 12-29, SCDS is your SMS source data set that is currently in use.

You see the IDG008I message as shown in Figure 12-30 after successful activation.

```
 RESPONSE=SC65      IGD008I NEW CONFIGURATION ACTIVATED FROM SCDS
 RESPONSE=SYS1.SMS.SCD
```

Figure 12-30 Activate SMS setting messages

Tailoring your SMS management class routines

Update your ACS routine to use an appropriate filter list and logic to assign management classes to data sets on tape. When a volume is opened, DFSMSrmm records the management class name you assign to new tape data sets using your ACS routine.

Example 12-1 shows a management class ACS routine that maps some of the special meaning expiration dates, for example 99000, to a management class name. If there is no match, a management class of CATALOG is set.

Example 12-1 Sample ACS management class routine

```
PROC 1 &MGMTCLAS
/*******************************************************/
/* DEFINE FILTER FOR TAPE UNITS                        */
/*******************************************************/
FILTLIST TAPEUNITS INCLUDE(TAPE*,T3420*,T3480*,T3490*,
                           '3420','3480','3490',
                           T3590*,'3590','3592')
/*******************************************************/
/* DATA SET NAME FILTLIST DEFINITIONS                  */
/*******************************************************/
 FILTLIST MCDB20   INCLUDE(DB2P.DSNDB%.*.%0*.**,
                           DB2T.DSNDB%.*.T0*.**,
                           DB2D.DSNDB%.*.T0*.**)
 FILTLIST MCDB21   INCLUDE(DB2P.DSNDB%.*.M1*.**,
                           DB2D.DSNDB%.*.T1*.**,
                           DB2T.DSNDB%.*.T1*.**)
 FILTLIST MCDB22   INCLUDE(DB2D.DSNDB%.*.T2*.**,
                           DB2T.DSNDB%.*.T2*.**)
   /*****************************************************/
   /* Support special meaning expiration dates          */
   /*****************************************************/
   IF &UNIT = &TAPEUNITS THEN DO
      WRITE 'MC ACS GETS CONTROL TO SET EXPDT MC'
      SELECT
         WHEN (&EXPDT = '1998001') SET &MGMTCLAS = 'M98001'
         WHEN (&EXPDT = '1999002') SET &MGMTCLAS = 'M98002'
         WHEN (&EXPDT = '1999003') SET &MGMTCLAS = 'M98003'
         WHEN (&EXPDT = '1990000') SET &MGMTCLAS = 'M99000'
         WHEN (&EXPDT = '1999000') SET &MGMTCLAS = 'M99000'
         WHEN (&EXPDT = '1999001') SET &MGMTCLAS = 'M99001'
         WHEN (&EXPDT = '1999002') SET &MGMTCLAS = 'M99002'
         WHEN (&EXPDT = '1999003') SET &MGMTCLAS = 'M99003'
         WHEN (&EXPDT = '1999004') SET &MGMTCLAS = 'M99004'
         OTHERWISE SET &MGMTCLAS = 'CATALOG'
      END
   END
   /*******************************************************/
   /* SELECT VALID MANAGEMENT CLASS                       */
   /*******************************************************/
   SELECT
      /*****************************************************/
      /* Accept special meaning expdt                      */
      /*****************************************************/
      WHEN (&MGMTCLAS = '' && &DSN = &MCDB20) SET &MGMTCLAS = 'MCDB20'
      WHEN (&DSN = &MCDB21) SET &MGMTCLAS = 'MCDB21'
      /*****************************************************/
      /* Accept any expdt                                  */
      /*****************************************************/
      WHEN (&EXPDT = '' && &DSN = &MCDB20) SET &MGMTCLAS = 'MCDB20'
```

```
   END
END
```

Important: If you allow the use of special meaning expiration dates in your computing center, specify the entire range of these dates in order not to lose data.

This example can be used for both system-managed and non-system-managed tape volumes. If you use the example for non-system-managed tape volumes only, you must include the code shown in Example 12-2.

Example 12-2 Sample ACS management class routine for tape only

```
IF &ACSENVIR = ’RMMVRS’ THEN
      SELECT
         WHEN (&EXPDT = ’1990000’) SET &MGMTCLAS = ’M99000’
         WHEN (&EXPDT = ’1998001’) SET &MGMTCLAS = ’M98001’
         ....
         WHEN (&EXPDT = ’1999000’) SET &MGMTCLAS = ’M99000’
         WHEN (&EXPDT = ’1999002’) SET &MGMTCLAS = ’M99002’
   ....
END
```

Tailoring your SMS storage group routines

To assign a storage group, you must have the SMS subsystem active, and have a valid SMS configuration.

You use the ACS routines to process the special calls that DFSMSrmm makes to the SMS subsystem for ACS processing. DFSMSrmm requests that the management class and storage group routines are run. The environment variable is set to `RMMPOOL` so that you can differentiate allocation requests for system-managed data sets from requests by DFSMSrmm for a storage group name. When DFSMSrmm calls the ACS routines with the &ACSENVIR variable, set to either `RMMPOOL` or `RMMVRS`; the storage class (&STORCLAS variable) is set to the word `USERMM`.

Important: You must use the EDGUX100 exit to request DFSMSrmm to disable the tape drive cartridge loader to prevent specific scratch pool requests from emptying the loaders through volume rejection. This function is not available using SMS ACS routines to select scratch pools.

Example 12-3 shows a simple example to use a specific scratch pool if the high-level qualifier is MHLRES1 or MHLRES4 for the new tape data set. Both volume pools must be defined in DFSMSrmm.

To make your decisions, you can use all read-only variables as described in 12.1.4, “SMS read-only variables” on page 446.

Example 12-3 ACS storage group routine for non-system-managed tape

```
IF &ACSENVIR = 'RMMVRS' THEN EXIT

IF &ACSENVIR = 'RMMPOOL' THEN DO
   WRITE 'SG ACS GETS CONTROL &ACSENVIR=' &ACSENVIR
   SELECT
      WHEN (&HLQ = 'MHLRES1') SET &STORGRP = 'PL3490'
      WHEN (&HLQ = 'MHLRES4') SET &STORGRP = 'LOAN'
   END
```

```
   EXIT
END
```

Tip: The SMS ACS management class routine is called first so that you can use this routine to make more powerful decisions in this routine and to use the result in the SMS ACS storage group routine.

Tailoring your EDGRMMxx PARMLIB member

Specify SMSACS(YES) in the EDGRMMxx PARMLIB member as shown in Example 12-4 to control whether DFSMSrmm calls SMS ACS processing to enable the use of storage group and management class values with DFSMSrmm.

Add new VLPOOL commands or update your previously defined VLPOOL commands in the EDGRMMxx PARMLIB member (see Example 12-4) to specify the NAME operand. If the NAME is a valid storage group name, it is used as the default STORGRP value for all volumes added into this pool range. You can choose another storage group name when adding a volume by using the STORGRP operand on the ADDVOLUME command.

The same NAME value can be used on multiple VLPOOL commands, enabling you to group multiple prefix ranges into a single logical pool without the need to specify STORGRP on ADDVOLUME.

By naming a specific storage group on ADDVOLUME, you can select, at the individual volume level, the pool to which it belongs.

Note: For manual tape libraries, DFSMSrmm uses the assigned storage group as the pool name to validate that a mounted scratch volume is from the requested pool.

Example 12-4 EDGRMMxx PARMLIB member to support SMS ACS processing

```
OPTION    OPMODE(R)                  /* Record-Only Mode          */  -
          SMSACS(YES)                /* enable MV ACS processing */   -
          .....
VLPOOL    PREFIX(VT*)                                                 -
          TYPE(R)                                                     -
          RACF(Y)                                                     -
          NAME(VTFM)                                                  -
          EXPDTCHECK(Y)                                               -
          DESCRIPTION('FOREIGN CARTRIDGES')
VLPOOL    PREFIX(L*)                                                  -
          TYPE(S)                                                     -
          RACF(Y)                                                     -
          NAME(LOAN)                                                  -
          EXPDTCHECK(Y)                                               -
          DESCRIPTION('TAPES FOR LOAN LOC PROCESSING')
```

To restart DFSMSrmm, use the modify command as shown in Figure 12-31 to implement updates to the data set, and to avoid stopping and restarting DFSMSrmm.

```
F DFRMM,M=xx
```

Figure 12-31 Restart DFSMSrmm to use the new parmlib settings

In Figure 12-31, *xx* is the suffix of the EDGRMMxx PARMLIB member.

You can also restart DFSMSrmm using another PARMLIB data set member with parameters to implement changes.

Figure 12-32 shows the messages after DFSMSrmm is successfully restarted.

```
EDG1101I DFSMSrmm MODIFY COMMAND ACCEPTED
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM03 IN RMM.PARMLIB
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

Figure 12-32 DFSMSrmm subsystem restart messages

Adding volumes to the volume pool

You can add new volumes to DFSMSrmm using the TSO RMM ADDVOLUME subcommand, or you can use the TSO RMM CHANGEVOLUME subcommand to update details for volumes that are already defined to DFSMSrmm. For both functions, you can also use the RMM ISPF dialog.

Use the RMM TSO ADDVOLUME subcommand that is shown in Figure 12-33 to add 10 new scratch volumes.

```
RMM ADDVOLUME LOAN01 COUNT(10) STATUS(SCRATCH) INIT(Y) POOL(VT*)
```

Figure 12-33 RMM ADDVOLUME subcommand

The following definitions apply to the command that is shown in Figure 12-33:

COUNT	Specifies the number of volumes to be added
STATUS	Specifies the volume status
INIT	Specifies whether the volume must be initialized before it can be used
POOL	Specifies a POOL where DFSMSrmm stores the volume in the removable media library

Note: When you do not specify a storage group name, DFSMSrmm assigns a storage group name by using the matching EDGRMMxx VLPOOL NAME value. If the VLPOOL NAME value is a valid storage group name, DFSMSrmm uses the VLPOOL NAME value as the default value for volumes added to the pool.

If you are using previously defined volumes, use the RMM TSO CHANGEVOLUME subcommand that is shown in Figure 12-34 to update some volume details.

```
RMM CHANGEVOLUME VT0001 POOL(VT*) STORAGEGROUP(VTFM)
```

Figure 12-34 RMM CHANGEVOLUME subcommand

In Figure 12-34, STORAGEGROUP specifies the SMS-defined storage group to which the volume belongs. POOL specifies a POOL where DFSMSrmm stores the volume in the removable media library.

Defining your VRSs

For all SMS management classes that you have defined and set in your SMS ACS routines, you must add a VRS in DFSMSrmm to define retention and movement policies. For more information about DFSMSrmm VRS processing, see Chapter 5, “Retaining and Moving Your Volumes” in *DFSMSrmm Guide and Reference*, SC26-7404.

A management class VRS must be defined as a data set VRS, and the data set name mask must match the management class name that is defined in SMS.

You can add new VRSs to DFSMSrmm by using the TSO RMM ADDVRS subcommand, or you can also use the RMM ISPF dialog.

Use the commands as shown in Figure 12-35 to define all new data set VRSs used in the sample ACS management class routine.

```
/* define VRSs for special meaning expdt processing 98001 to 99364  */
RMM ADDVRS DATASETNAME('M98001') COUNT(1) LASTREFERENCEDAYS
RMM ADDVRS DATASETNAME('M98002') COUNT(2) LASTREFERENCEDAYS
RMM ADDVRS DATASETNAME('M98003') COUNT(2) LASTREFERENCEDAYS
/* define VRSs for special meaning expdt processing 90000 and 99000 */
RMM ADDVRS DATASETNAME('M99000') CYCLES COUNT(99999) WHILECATALOGED
/* define VRSs for special meaning expdt processing 99001 to 99364  */
RMM ADDVRS DATASETNAME('M99001') COUNT(1) CYCLES
RMM ADDVRS DATASETNAME('M99002') COUNT(2) CYCLES
RMM ADDVRS DATASETNAME('M99003') COUNT(3) CYCLES
RMM ADDVRS DATASETNAME('M99004') COUNT(4) CYCLES
/* define VRSs for all other used management classes                */
RMM ADDVRS DATASETNAME('MCDB20') COUNT(20) BYDAYCYCLES WHILECATALOG -
    UNTILEXPIRED SCRATCHIMMEDIATE NEXTVRS(SAVE)
RMM ADDVRS NAME(SAVE) LOCATION(SAVE) COUNT(5)
RMM ADDVRS DATASETNAME('MCDB21') COUNT(21) CYCLES WHILECATALOG -
    UNTILEXPIRED SCRATCHIMMEDIATE NEXTVRS(EXTRA)
RMM ADDVRS NAME(EXTRA) LOCATION(CURRENT) EXTRADAYS(2)
```

Figure 12-35 Defining new data set name VRSs

12.1.6 Migration considerations

If you are already using DFSMSrmm, use this new function to migrate your EDGUX100 management value and volume pool selection functions to the DFSMSrmm SMS ACS support.

Move pooling decisions out of the EDGUX100 user exit

You can implement DFSMSrmm scratch pooling by using ACS routines based on storage group names. DFSMSrmm provides support for non-system-managed tape and for system-managed manual tape libraries. This support enables pooling at the individual volume level. You assign a storage group name to each volume by one of these methods:

- Using DFSMSrmm TSO subcommands
- Using pool information that you define with the DFSMSrmm EDGRMMxx PARMLIB VLPOOL command

You use the ACS routines to process the special calls that DFSMSrmm makes to the SMS subsystem for ACS processing. DFSMSrmm requests that the management class and storage group routines are run.

Moving VRS management decisions out of EDGUX100 user exit

You use the ACS routines to process the special calls that DFSMSrmm makes to the SMS subsystem for ACS processing. In the replacement of the EDGUX100 user exit VRS management value, DFSMSrmm requests that the management class routine is run.

For each new tape data set, DFSMSrmm processing follows the steps described here depending on the circumstances (see Figure 12-36):

- Pre-ACS processing (for details, see “Pre-ACS support as a way to migrate” on page 467):
  - The IGDACSXT exit is called. You can implement this exit to set values to the &MSPOLICY, &MSDEST, &MSPOOL, and &MSPARM ACS read-only variables before ACS routines are called. Then, your ACS routines can refer to these variables.
  - DFSMSrmm calls the EDGUX100 installation exit to enable a VRS management value to be returned. The selected value is returned in the &MSPOLICY variable.
- OPEN processing after a device has been allocated:
  - DFSMSrmm calls SMS ACS processing to process the management class ACS routine passing environment information.
  - If a management class is not returned from ACS processing, DFSMSrmm calls the EDGUX100 installation exit to enable a VRS management value to be returned.
  - The specified management class or VRS management value is recorded by DFSMSrmm at the data set level.

> **Important:** The environment variable &ACSENVIR is set to RMMVRS so that you can differentiate allocation requests for system-managed data sets from requests by DFSMSrmm for a management class name.

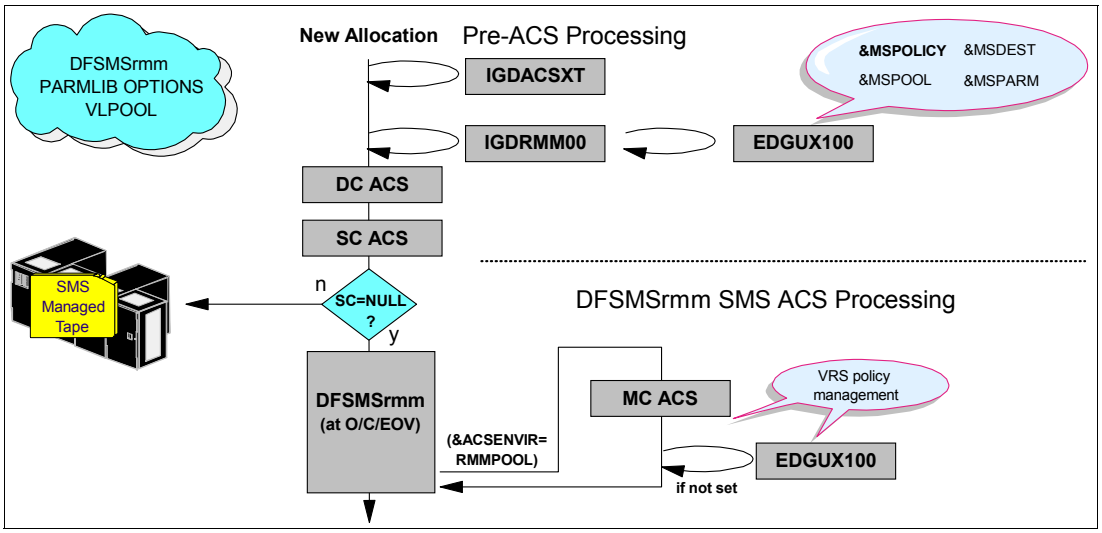


Figure 12-36 DFSMSrmm V2R10 SMS ACS VRS policy management support

Pre-ACS support as a way to migrate

Using the SMS pre-ACS calls to DFSMSrmm, you can use any existing DFSMSrmm and EDGUX100 exit scratch pool and EDGUX100 exit policy assignment decisions as input to your SMS ACS logic. This capability enables you to direct new data set allocations to the correct media. You can use SMS ACS to decide whether data sets are to be system managed or non-system managed. For system-managed tape data sets, you can decide to which library or library type you want the allocation directed, which can help you with Virtual Tape Server (VTS) implementation.

You can use the pre-ACS interface to provide additional information, such as vault destination or pool information, to the ACS routines. You can use the EDGUX100 installation exit for pre-ACS processing. When called during pre-ACS processing, values selected by the exit are used as input to the ACS routine.

DFSMSrmm always attempts to pass values for the &MSPOOL and &MSPOLICY ACS read-only variables. If you do not use an EDGUX100 exit, the &MSPOOL variable is set to the DFSMSrmm system-based scratch pool decision. If you do not use an EDGUX100 exit, there is never a variable set for &MSPOLICY. DFSMSrmm sets a pre-ACS variable only if the variable has not yet been set using the pre-ACS exit. Since DFSMSrmm gets control after the installation exit, any vendor or customer decisions take precedence.

DFSMSrmm can set the following ACS read-only variables:

- Pool name **&MSPOOL**

  Used to specify a tape pool name associated with a data set that is being allocated. In a system managed tape library, this variable might be used to specify a default storage group where the storage group is equivalent to the tape pool that is specified in the library.

- Policy name **&MSPOLICY**

  Used to specify a management policy that is related to a system-managed library

If you only want to use &MSPOLICY and &MSPOOL variables, it is enough to code the EDGUX100 installation exit. EDGUX100 gets control right after IGDACSXT returns the control to the system, and it sets the values only if IGDACSXT did not set them.

In this way, you can use EDGUX100 assignment decisions as input to your SMS ACS logic to enable you to direct new data set allocations to the correct media. The pre-ACS routine passes the address of the ACS installation exit control block ACERO to the EDGUX100 installation exit. The ACERO is mapped by the IGDACERO macro and is the input to the pre-ACS installation exit IGDACSXT. For more information about how you can use the ACERO, see *z/OS DFSMS Installation Exits*, SC26-7396.

You can use any of the values in the ACERO as input to the EDGUX100 installation exit. The sample EDGUX100 exit uses these values:

ACEROJOB	What is the job name?
ACERODSN	What is the data set name?
ACEROEXP	What is the expiration date? The expiration date is used only if the retention period is not set.
ACERORTP	What is the retention period?

The values are used to perform PL100_CAN_IGNORE, PL100_CAN_VRS, and PL100_CAN_POOL processing. All other functions are performed only when the EDGUX100 is not called by the pre-ACS routine.

If your installation's version of the EDGUX100 installation exit provides values for scratch pooling (PL100_POOL) or a VRS management value (PL100_VRS), DFSMSrmm updates the ACS input values for MSPOOL and MSPOLICY when they have not already been set by the IGDACSXT pre-ACS installation exit.

Deciding whether to use pre-ACS or DFSMSrmm SMS ACS processing

Our intention is to enable you to move function out of the EDGUX100 installation exit. We advise that you do not use the PREACS option in the EDGRMMxx PARMLIB member.

You can use the existing decisions that are coded in your EDGUX100 exit to input into your SMS ACS logic. Within system-managed tape, you can decide to which library or library type you want the allocation to be directed. This can help with VTS implementation. Do this by using the pre-ACS processing.

Implement or customize the sample EDGUX100 exit (or the EDGCVRSX sample of EDGUX100) to feed pool and VRS management value information into ACS. Base your ACS processing decisions on the MSPOOL and MSPOLICY values that come from DFSMSrmm scratch pool and EDGUX100 exit processing. You can plan to move the EDGUX100 exit decisions into your SMS ACS routines to enable the EDGUX100 exit processing to be ignored or removed.

By using SMS ACS processing, you can consolidate scratch pooling and policy assignment decisions in a single place whether you use system-managed tape or not.

12.2 Using DFSMSrmm with virtual tape solutions

Virtual tape solutions are always related to the use of an automated tape library, whether it is system managed or non-system managed. Therefore, virtual tape solutions have a different way of working than the traditional tapes. This is why we want a separate section to explain how they work, and how DFSMSrmm provides special support for these solutions and their specific functions.

The DFSMSrmm support for virtual solutions is basically the same for IBM Virtual Tape Server and for other implementations, but there are some specifics for VTS. The interface between DFSMSrmm and OAM is built in and automated; the interface for other virtual tape solutions must be driven by DFSMSrmm commands.

DFSMSrmm supports virtual tape server import/export in different ways, depending on whether DFSMSrmm stacked volume support is enabled. When you enable stacked volume support, DFSMSrmm tracks logical volumes and stacked volumes.

To verify that the stacked volume support is enabled, go to the DFSMSrmm Control Record Display panel shown in Figure 12-37.

```
                        DFSMSrmm Control Record Display
Command ===>
                                                                More:     +
Type  . . : MASTER      Create date . : 2009/309    Create time . : 18:38:11
                        Update date . : 2010/291    Update time . : 12:55:05

Journal utilization . :   0 %    ( 75 % threshold)  Status : ENABLED
CDS utilization . . . :  19 %

Exit status:                            Options:
EDG_EXIT100 . . : ENABLED                  Stacked volumes . . . : NONE
EDG_EXIT200 . . : ENABLED                  Extended bin  . . . . : ENABLED
EDG_EXIT300 . . : ENABLED                  Common time . . . . . : DISABLED
```

Figure 12-37 DFSMSrmm Control Record Display panel

If it is not enabled, use the EDGUTIL program, as shown in Figure 12-38 on page 470.

```
//CREATE   EXEC PGM=EDGUTIL,PARM='UPDATE'
//MASTER    DD  DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  CONTROL CDSID(PROD) STACKEDVOLUME(YES)
/*
```

Figure 12-38 EDGUTIL example

12.2.1 VTS support for stacked volumes

Before OS/390 R10 DFSMSrmm, VTS support was limited to a new logical volume type, with the ability to record the container volume for any volume defined to DFSMSrmm. Stacked volumes are not defined or managed as a stacked volume for DFSMSrmm.

With OS/390 V2R10, DFSMSrmm supports stacked volumes defined to DFSMSrmm. If you do not define stacked volumes, DFSMSrmm defines them automatically at export time if stacked volume support is enabled.

To define stacked volumes, you can use ISPF panels or the RMM TSO subcommand shown in Figure 12-39.

```
RMM ADDVOLUME STO000 TYPE(STACKED) STATUS(MASTER) LOCATION(LIBVTS)
```

Figure 12-39 RMM ADDVOLUME subcommand to add a stacked volume

Note: You cannot remove stacked volume support after it is enabled. We advise that you enable stacked volume support whether or not you have a VTS.

When you define a stacked volume, DFSMSrmm checks the TCDB and the library manager. If the volume is not known to the library manager, the volume location is set to `SHELF` and the destination is set to the VTS library name.

Figure 12-40 shows the VTS support for stacked volumes.

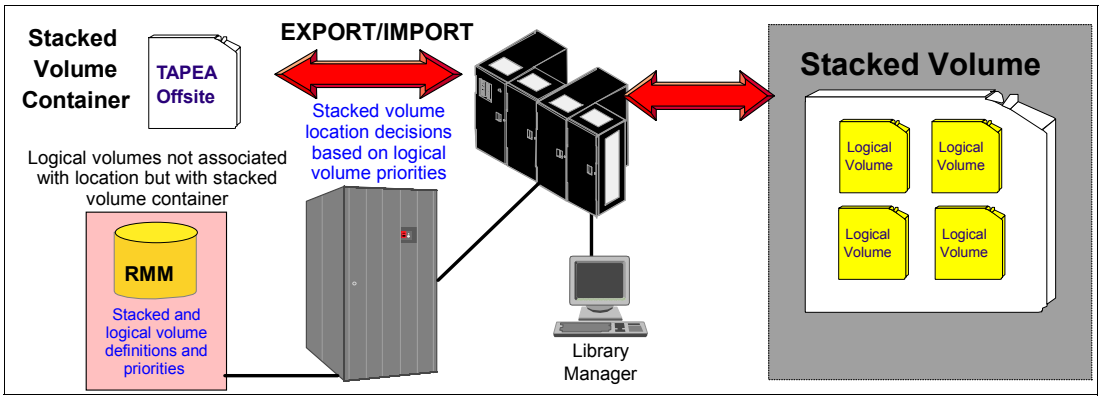


Figure 12-40 VTS support for stacked volumes

At export time, the DFSMSrmm control data set shows that logical volumes are on stacked volumes (specified in the In container field for each logical volume), and stacked volumes are known and managed by DFSMSrmm.

The movement of the stacked volumes is specified for the logical volumes. DFSMSrmm drives the stacked volume movement based on the required location of the contained volume.

Export processing

This example explains how to export a data set IBMUSER.TAPE3490.PRUACS on logical volume MWW000 without previously defining the stacked volume:

1. Define the VRS for the data sets that you want exported as shown in Figure 12-41.

```
RMM AS DSNAME('IBMUSER.TAPE3490.**') DAYS COUNT(3) LOCATION(REMOTE)
```

Figure 12-41 RMM ADDVRS subcommand

2. Run inventory management VRSEL processing to determine which logical volumes need to be moved and to set the required location for each volume. For our volume, you see the information in Figure 12-42.

```
Location:    Current      Destination  Old          Required     Home
Name       = LIB2         REMOTE                                LIB2
Type       = AUTO         STORE                                 AUTO
```

Figure 12-42 VRSEL processing output

3. Figure 12-43 shows how to create a list of volumes to export.

```
RMM SV VOLUME(*) LIMIT(*) REQUIRED(REMOTE) LOCATION(ATLBA999) -
    CLIST('','' REMOTE')
```

Figure 12-43 RMM SEARCHVOLUME subcommand

4. Start the export process as described in the *DFSMS OAM Planning, Installation, and Storage Administrator Guide for Tape Libraries*, SC35-0427. During this process, DFSMSrmm updates the field “in container” in the logical volume. The stacked volume DD0813 is added to DFSMSrmm, with a volume type of STACKED.
5. Run inventory management DSTORE processing to set the destination for the volume.
6. Eject the stacked volume from the VTS from the Library Manager.
7. Confirm that the stacked volume was moved by using the command shown in Figure 12-44.

```
RMM CHANGEVOLUME DD0813 CONFIRMOVE
```

Figure 12-44 RMM CHANGEVOLUME subcommand to confirm a volume move

Import processing

DFSMSrmm supports importing logical volumes that might be defined to DFSMSrmm. During import processing for a logical volume, DFSMSrmm automatically adds the volume information to the DFSMSrmm control data set or leaves the volume to be processed on another system.

The following steps explain how to import a volume:

1. Use the library manager to transfer a stacked volume from the “un-assign” category to the “import” category:

   a. Create an import list. You can create this list by using stacked volumes or logical volumes. To build a list of stacked volumes, use the command shown in Figure 12-45.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) TYPE(STACKED)-
    DESTINATION(lib_vts) CLIST
```

Figure 12-45 RMM SEARCHVOLUME subcommand for a single destination

   b. To build a list of logical volumes to import from a single stacked volume, use the command that is shown in Figure 12-46.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(*) CONTAINER(DD0813)-
    TYPE(LOGICAL) CLIST
```

Figure 12-46 RMM SEARCHVOLUME subcommand for a single container volume

   c. To build a list of logical volumes to import from multiple stacked volumes, use the command that is shown in Figure 12-47.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(*) CONTAINER(*)-
    TYPE(LOGICAL) REQUIRED(lib_vts) CLIST
```

Figure 12-47 RMM SEARCHVOLUME subcommand with the required operand

2. Start the import process. If the volume is already known to DFSMSrmm, it is accepted for processing if it is defined as an exported logical volume; otherwise, it is rejected. During this process, DFSMSrmm removes the stacked volume association from the logical volume.

Table 12-2 shows the information that DFSMSrmm tracks in import and export processing for stacked volumes.

Table 12-2 Stacked volume information

Field	Export	Import
Status	NOT EMPTY	EMPTY
Assign Date	Set at start	Set at end
Stacked Count	Increment	Decrement
Use Count	Increment at start	
Usage	Sum of contained volumes	Sum of contained volumes
Security Level	Highest	
Last Read Date		Set
Last Write Date	Set	

You can use DFSMSrmm logical and stacked volumes together with DFSMSrmm commands for *any* virtual tape solution. See 12.2.2, “Stacked volume support for non-VTS” on page 473.

12.2.2 Stacked volume support for non-VTS

You can use stacked volume support for virtual tape solutions that are different from VTS. The difference is that you must provide to DFSMSrmm the information that OAM provides for VTS during export and import processing.

In this case, you must define to DFSMSrmm stacked and virtual volumes using the RMM commands that are shown in Figure 12-48.

```
RMM ADDVOLUME stacked_vol TYPE(STACKED) LOCATION(SHELF)
RMM ADDVOLUME logical_vol TYPE(PHYSICAL/LOGICAL) -
    STATUS(MASTER) CONTAINER(stacked_vol)
```

Figure 12-48 RMM ADDVOLUME subcommands

DFSMSrmm is notified by OAM for import and export processing, and it can obtain movement completion information from the Library Manager. You must perform these functions by using commands.

If you have already defined virtual and stacked volumes, you can add a virtual volume to a container by using CV CONTAINER(*name*). CONTAINER(' ') removes the logical volume from the container.

Figure 12-49 shows how DFSMSrmm manages stacked, container, and logical volumes in the export process.

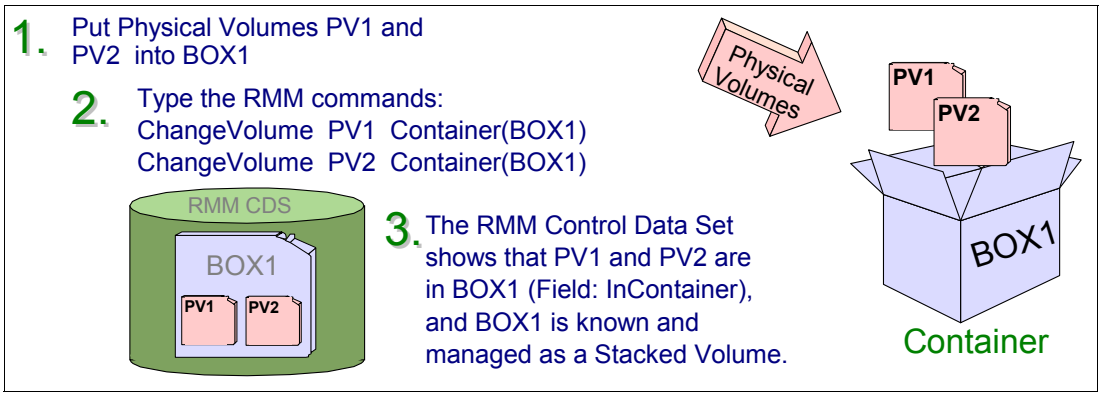


Figure 12-49 Export process for non-VTS

13

System Authorization Facility tape security

z/OS Data Facility System Managed Storage (DFSMS) V1.8 or higher provides new options for securing tape data sets by using System Authorization Facility (SAF). These options are designed to allow you to define profiles to protect data sets on tape by using the DATASET class without needing to activate the TAPEDSN option or the TAPEVOL class. DFSMS also provides options that you can use to specify that all data sets on a tape volume have common authorization and that users are authorized to overwrite existing files on a tape volume.

In this chapter, we describe the options available in z/OS DFSMS V1.8 or higher for tape data set security and how to implement it.

This chapter contains the following topics:

- Tape data set authorization
- Implementation
- Removing TAPEVOL and TAPEDSN processing
- Error messages
- Testing various security settings

13.1 Tape data set authorization

z/OS DFSMS V1.8 or higher provides options for securing tape data sets using the system authorization facility (SAF), to allow you to protect data sets on tape using the RACF DATASET class without needing to activate the TAPEDSN option or the TAPEVOL class. This new tape data set authorization checking allows you to specify that all data sets on a tape volume should have common authorization. You can specify whether users are authorized to overwrite existing files on a tape volume. In addition, you can specify that a user must have access to the first file on a tape volume to add additional files on that tape volume.

13.1.1 Suggestions for tape security

For optimum tape security, use the combined capabilities of DFSMSrmm, DFSMSdfp, and RACF. It is advised that you specify the following parameters:

- In the DEVSUPxx parmlib member:
  - TAPEAUTHDSN=YES
  - TAPEAUTHF1=YES
  - TAPEAUTHRC4=FAIL
  - TAPEAUTHRC8=FAIL
- In the EDGRMMxx parmlib member:
  - OPTION TPRACF(N)
- In RACF:
  - SETROPTS NOTAPEDSN NOCLASSACT(TAPEVOL)

The combination of DFSMSrmm, DFSMSdfp, and RACF helps to ensure the following functions:

- Full 44-character data set name validation.
- Validation that the correct volume is mounted.
- Control over the overwriting of existing tape data sets.
- Management of tape data set retention.
- Control over the creation and destruction of tape volume labels.
- No limitations caused by RACF TAPEVOL profile sizes and tape volume table of contents (TVTOC) limitations.
- All tape data sets on a volume have a common authorization.
- Use of generic DATASET profiles, enabling common authorization with DASD data sets.
- Authorization for all tape data sets, regardless of the tape label type.
- Authorization for the use of bypass label processing (BLP).
- Use of RACF erase on scratch support.
- Use of DFSMSrmm FACILITY class profiles for data sets unprotected by RACF. Your authorization to use a volume outside of DFSMSrmm control through ignore processing also enables authorization to the data sets on that volume.

To aid in the migration to this suggested environment, DFSMSrmm provides the TPRACF(CLEANUP) option, and DEVSUPxx provides TAPEAUTHRC8(WARN) and TAPEAUTHRC4(ALLOW).

The function in DFSMSdfp does not replace all the functional capabilities that the RACF TAPEDSN option, TAPEVOL class, and TVTOC provide. However, together with the functions that DFSMSrmm provides, you have equivalent capability.

The enhanced DFSMSdfp function addresses the authorization requirements for tape data sets and relies on your use of a tape management system, such as DFSMSrmm, to perform the following operations:

- Verify full 44-character data set names.
- Control the overwriting of existing tape files.
- Handle tape data set retention.
- Control the creation and destruction of tape labels.

Important: Tape volume profiles that contain a TVTOC have following restrictions:

- The maximum number of entries for data sets that a TVTOC can contain is 500.
- The maximum number of volumes that any data set on the tape with an entry in the TVTOC can span is 42.
- RACF does not permit duplicate data set names on the same volume, or on different volumes if the volumes are part of the same multivolume data set group.
- A TAPEVOL profile can span a maximum of 10,000 tape volumes.
- You cannot delete a tape volume from a tape volume set if a TVTOC entry indicates that there is a protected data set on the volume.

13.1.2 Overview of TAPEVOL and TAPEDSN processing

The enhancement in authority checking that is available with z/OS DFSMS V1.8 or higher takes into account that today most clients are using a tape management system. A tape management system, such as DFSMSrmm, verifies during reuse of data sets and tape volumes the data set names that were put in place at creation time. This ensures that the same data set resources are always being checked in RACF whenever a particular data set is opened.

How protection of tape data sets works before z/OS DFSMS V1.8

Processing in detail differs depending on these factors:

- Setting of SETROPTS TAPEDSN/NOTAPEDSN
- RACF TAPEVOL class being active or not
- Label type of the tape volume
- Requirement of BLP

SETROPTS NOTAPEDSN NOCLASSACT(TAPEVOL)

Tape data sets are not protected if neither RACF option TAPEDSN is being set nor the RACF TAPEVOL class is active. This is true regardless of label type or BLP requirement.

SETROPTS TAPEDSN and SETROPTS CLASSACT(TAPEVOL)

If the RACF TAPEVOL class is active as well as the RACF option TAPEDSN being set, tape data is protected at the data set level.

In the case of label type SL or AL, during the open of an existing data set, a REQUEST=AUTH is performed in the DATASET class with DSNTYPE=T:

1. If there is a matching TAPEVOL class profile, the caller's authority according to the type of open is being checked (READ for input and UPDATE for output):
   - If the caller's authority at the volume level is sufficient, no further checks are performed.
   - Otherwise, if the TAPEVOL profile contains a TVTOC, the data set name is verified at the full length of 44 bytes:
     - If the verification was successful, RACF looks for a matching data set profile and determines whether the caller's authority is sufficient.
     - If verification was not successful, the request ends with return code 8 (denied).
   - If the TAPEVOL profile contains no TVTOC, see step 2.
2. Otherwise, if there is no matching TAPEVOL profile, RACF looks for a matching DATASET class profile and determines whether the caller is authorized sufficiently (READ for input and UPDATE for output).

Note: Without a TVTOC, there is no verification of the full 44-byte length of the data set name.

With a label type of SL or AL, during the open of a new data set, a REQUEST=AUTH, as well as a REQUEST=DEFINE, is performed in the DATASET class with DSNTYPE=T:

- If the caller wants to append another data set:
  a. If there is a matching TAPEVOL class profile, the caller must have at least UPDATE authority; otherwise, the request ends with return code 8 (denied).
  b. RACF is looking for a matching data set profile and determines that the caller is at least authorized for UPDATE access.
- If the caller wants to overwrite an existing data set:
  - If the new data set name is different, the existing data set, either the security retention period of that particular existing data set (stored in the protecting DATASET profile) and all of the following existing data sets must have expired:
    - Or, the caller must have at least UPDATE authority to the volume.
    - RACF looks for the matching profile of the new data set name and determines that the caller is at least authorized for UPDATE access.
  - If the new data set name is *not* different, the caller must have UPDATE authority:
    - Either for the matching DATASET profile
    - Or for the matching TAPEVOL profile
- If the TAPEVOL profile contains a TVTOC, an entry in the TVTOC is maintained for the new data set.

Note: When a TAPEVOL class profile contains a TVTOC, several restrictions apply:

- The maximum number of entries for data sets that a TVTOC can contain is 500.
- The maximum number of volumes that any data set on the tape with an entry in the TVTOC can span is 42.
- RACF does not permit duplicate data set names on the same volume or on different volumes if the volumes are part of the same multivolume data set group.
- A TAPEVOL profile can span a maximum of 10,000 tape volumes.
- From a tape volume set, you cannot delete a tape volume if a TVTOC entry indicates that there is a protected data set on the volume.

In the case of non-labeled tapes (NL):

- During the open for input, the caller must have READ authority either to the volume or, if there is a TVTOC for the volume, to the matching data set profile.
- During the open for output, the caller must have at least UPDATE authority to both the volume and the data set.

In the case of nonstandard label tapes (NSL), data management does not perform authorization checking.

In the case of a request for BLP, during the opening of a data set, the caller must be authorized for the ICHBLP resource in the FACILITY class.

SETROPTS TAPEDSN and SETROPTS NOCLASSACT(TAPEVOL)

If only the RACF option TAPEDSN is being set while the TAPEVOL class is not active, tape data is protected at the data set level.

In the case of label type SL, AL, or NL during the opening of a data set for reading, the caller needs at least READ authority for the matching DATASET class profile.

In the case of label type SL, AL, or NL during the opening of a data set for writing, the caller needs at least UPDATE authority to open an existing data set or ALTER authority to create a new data set for the matching DATASET class profile.

In the case of nonstandard label tapes (NSL), data management does not perform authorization checking.

In the case of a request for BLP during the opening of a data set, the caller must be authorized for the ICHBLP resource in the FACILITY class.

SETROPTS NOTAPEDSN and SETROPTS CLASSACT(TAPEVOL)

If the RACF option TAPEDSN is not being set while the TAPEVOL class is active, tape data is protected at the tape volume level.

In the case of label type SL, AL, or NL, during the opening of a data set for reading, the caller needs at least READ authority for the matching TAPEVOL class profile.

In the case of label type SL, AL, or NL, during the opening of a data set for writing, the caller needs at least UPDATE authority for the matching TAPEVOL class profile.

In the case of nonstandard label tapes (NSL), data management does not perform authorization checking.

In the case of a request for BLP during the opening of a data set, the caller must be authorized for the ICHBLP resource in the FACILITY class.

Summary

To achieve the highest level of security, choose an environment that provides discrete TAPEVOL class profiles with TVTOCs, as well as the SETROPTS TAPEDSN setting. This environment provides the following advantages:

- Full 44-character data set name validation
- The use of generic DATASET class profiles, enabling common authorization with DASD data sets
- Volume level control to help guarantee exclusive volume use for applications

13.1.3 How the SAF tape data set authority checking works

The alternate authority checking is enabled by modifying the contents of member DEVSUPxx in the parmlib concatenation. Four new keywords are available:

- **TAPEAUTHDSN**:

  YES Enables tape authorization checks in the DATASET class but without DSTYPE=T. The system uses the data set name specified in the allocation or JCL to check your authorization to read or write the specified file. In addition, the system determines the RACF erase-on-scratch setting from the RACF profile and passes it to your tape management system. When you request BLP and the mounted volume uses standard labels, OPEN issues the authorization check that the user is authorized to use BLP. This processing uses the existing ICHBLP resource in the RACF FACILITY class. When you specify TAPEAUTHDSN = YES only, it replaces the check that RACF makes as part of tape volume authorization checking.

  NO Indicates that OPEN processing is to issue RACROUTEs as it did before based on the options set in RACF, such as SETROPTS TAPEDSN and SETROPTS CLASSACT(TAPEVOL).

  This is the default setting.

- **TAPEAUTHF1**:

  YES Enables additional tape authorization checks in the DATASET class for existing files on the same tape volume when any other file on the tape volume is opened. This function depends on the tape management system returning the 44-character data set name and data set sequence number to OPEN/end of volume (EOV) through the IFGTEP during the Volume Mount exit Volume Security function. If no data set name is returned by the tape management system, processing occurs as though this keyword is not specified. Although intended to enable an additional authorization check for the first data set when any other data set on the tape volume is opened, the implementation allows your tape management system to request one or more additional authorization checks when any data set on a tape volume is opened. Each additional data set name and data set sequence number returned results in an additional RACROUTE. Do not use this function unless you have a tape management system and it can return a data set name and

data set sequence number. A *data set sequence number* is the label number normally specified in the JCL LABEL keyword and stored in the catalog. When TAPEAUTHDSN=YES is in use, any additional RACROUTE matches that issued for TAPEAUTHDSN, except for the data set name and data set sequence number. Otherwise, TAPEAUTHF1 uses a RACROUTE that matches that used for SETROPTS TAPEDSN. When neither TAPEAUTHDSN nor SETROPTS TAPEDSN is in use, TAPEAUTHF1 support is not provided.

NO Disables additional tape authorization checks in the DATASET class for existing files on the same tape volume when any other file on the tape volume is opened. This is the default setting.

- **TAPEAUTHRC4**:

  ALLOW This applies to authorization checks in the DATASET class, and applies only to the results of TAPEAUTHDSN = YES and TAPEAUTHF1 = YES processing. Allows the access of data sets that are not protected by a security profile. RC4 refers to the return code value of 4 returned from SAF as a result of the RACROUTE issued by OPEN/CLOSE/EOV. A return code of 4 generally means that the resource is not protected.

  FAIL Denies the access of data sets that are not protected by a security profile. Use this setting in a PROTECTALL(FAIL) environment. This is the default setting.

- **TAPEAUTHRC8**:

  WARN Enables warning mode for all tape authorization checks in the DATASET class as a result of TAPEAUTHDSN = YES and TAPEAUTHF1 = YES processing. Allows the access of data sets that typically cannot be accessed. RACF issues an ICH408I message to indicate why access is not allowed; however, OPEN/EOV allows access.

  FAIL Denies the access of data sets according to the result of the check in the DATASET class. This is the default setting.

In Table 13-1, you can see the different PROTECTALL settings in RACF and the equivalent settings of the new TAPEAUTHRC4 and TAPEAUTHRC8 options for securing tape data sets using SAF.

Table 13-1 Compare RACF and DEVSUP protection

RACF settings	DEVSUP settings	Action
PROTECTALL(NONE)	N/A	None
PROTECTALL(WARN)	TAPEAUTHRC4(ALLOW) TAPEAUTHRC8(WARN)	Enables warning mode
PROTECTALL(FAIL)	TAPEAUTHRC4(FAIL) TAPEAUTHRC8(FAIL)	Denies the access of data sets: ► That are not protected ► According to the result of the check

Figure 13-1 shows the RACF checking sequence when you set the new tape security options TAPEAUTHDSN and TAPEAUTHF1 and activated them. If you have not specified to bypass DFSMSrmm processing by using the JCL parameter EXPDT=98000 or ACCODE=XCANORES, the RACF checking will occur. The data sets must be RACF protected and the user must have access to the data set profile protecting the data sets.

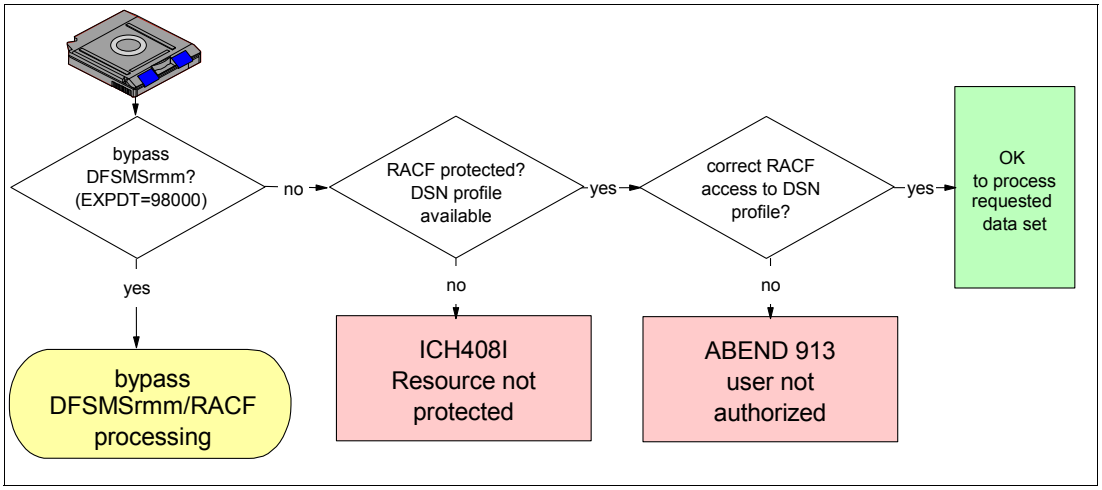


Figure 13-1 Tape security flowchart

If you specified to bypass DFSMSrmm processing by using the JCL parameter EXPDT=98000 or ACCODE=XCANORES, DFSMSrmm will perform several additional security checks as shown in Figure 13-2 on page 483. Instead of using different RACF resources to check the bypass processing for volumes defined or not defined in the DFSMSrmm control data set, you can specify the resource STGADMIN.EDG.IGNORE.TAPE.* in the FACILITY class.

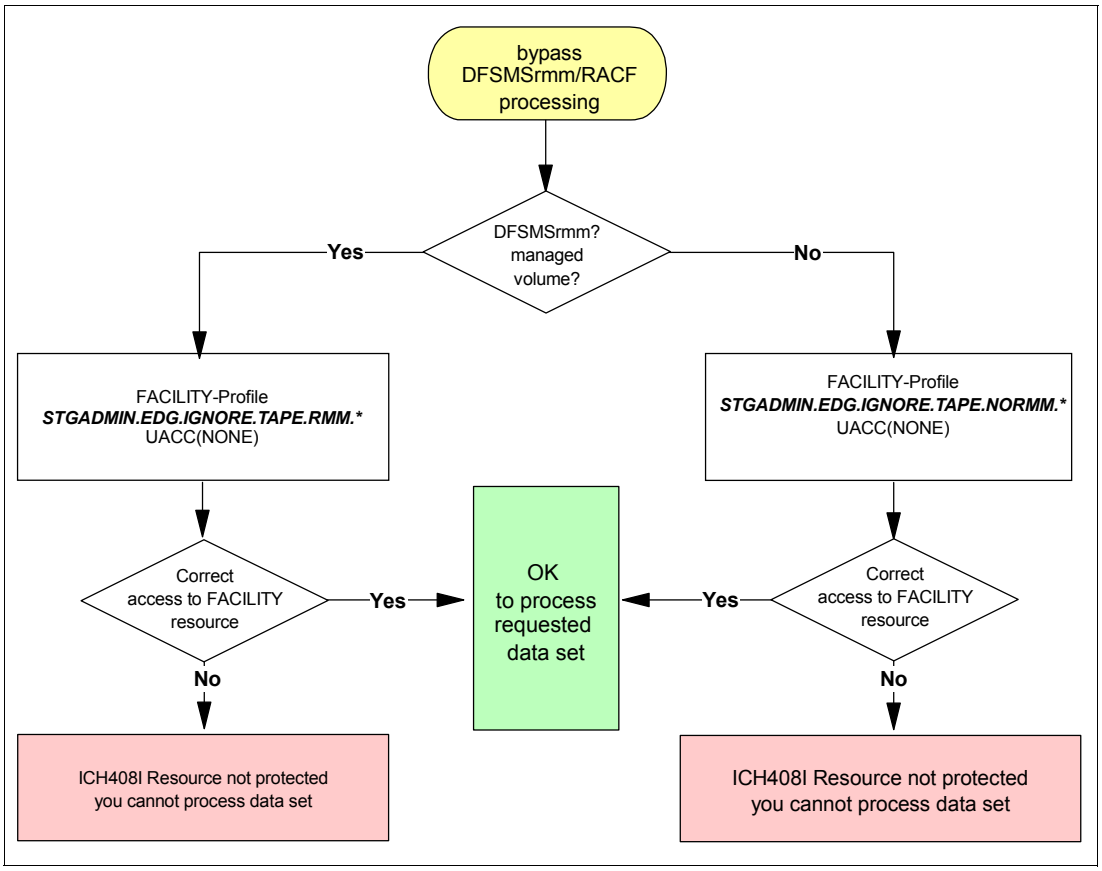


Figure 13-2 Bypass DFSMSrmm processing flowchart

Note: You can specify a profile in the RACF class FACILITY for each volume by using the full volume serial number at the end of the resource name, such as STGADMIN.EDG.IGNORE.TAPE.RMM.**V12345**. Or, you can specify a profile in the RACF class FACILITY for a range of volumes using the asterisk as part of the volume serial number, such as STGADMIN.EDG.IGNORE.TAPE.NORMM.**ABC***, for all three kinds of resources.

13.2 Activating RACF TAPEVOL and TAPEDSN

Instead of implementing RACF TAPEVOL and TAPEDSN, you can protect your data sets on tape by using the SAF tape security introduced with DFSMS z/OS V1R8. For more information, see Chapter 13, “System Authorization Facility tape security” on page 475.

You can control the actions that DFSMSrmm takes on RACF tape profiles through the DFSMSrmm OPTION TPRACF command and your installation’s RACF options. To protect tape volumes, you can specify DFSMSrmm OPTION TPRACF(AUTOMATIC) or TPRACF(PREDEFINED). For more information, see 13.2.4, “DFSMSrmm OPTION command operand TPRACF” on page 485.

Before you execute the RACF commands, activate the RACF TAPEVOL and TAPEDSN classes as shown in Figure 13-3 on page 484.

```
SETROPTS CLASSACT(TAPEVOL)
SETROPTS TAPEDSN
SETROPTS LIST
```

Figure 13-3 RACF SETROPT commands

Suggestion: We advise that you do not activate the RACF TAPEVOL and TAPEDSN classes until after you convert to and successfully use DFSMSrmm for a period of time (for example, three months).

For optimal security, we suggest that you make TAPEVOL and TAPEDSN active with either EDGRMMxx PARMLIB option TPRACF(P) or TPRACF(A) to obtain full protection at the volume and data set level.

Although we discourage this approach, it is possible to have no tape protection, or to use only TAPEVOL profiles. If you have no tape security, you cannot control the creation and use of data on tape. If you use only TAPEVOL profiles to protect tape volumes, you must maintain the access lists to the TAPEVOL profiles to allow access. Consider the use of TAPEDSN so that access to tape data is covered by your normal, existing, generic data set profiles.

13.2.1 Before you can start

You can implement RACF TAPEVOL and TAPEDSN only if you have a user with all of the required authority and access.

Your user ID needs the following attributes to execute all created RACF commands:

- **SPECIAL**: Specifies that the user is allowed to issue all RACF commands with all operands except the operands that require the AUDITOR attribute.
- **AUDITOR** (optional): Specifies that the user has full responsibility for auditing the use of system resources. An AUDITOR user can control the logging of detected accesses to any RACF-protected resources during RACF authorization checking and access to the RACF data set.

You need access to the following data set profiles and resources in the FACILITY class that is defined in RACF to create a new report extract data set:

- ALTER access to data set profile RMM.HSKP.**
- UPDATE access to data set profile SYS1.PARMLIB or any equivalent PARMLIB in which you have defined the EDGRMMxx member
- READ access to STGADMIN.EDG.HOUSEKEEP defined in the FACILITY class
- READ access to STGADMIN.EDG.HOUSEKEEP.RPTEXT defined in the FACILITY class

13.2.2 DFSMSrmm RACF tape security support

The objective for DFSMSrmm RACF tape security support is to provide a complete interface with RACF for tapes, so that you can use any combination of valid RACF options, including the use of any RACF installation exits you still have. You can tell DFSMSrmm not to provide any RACF support by specifying the DFSMSrmm EDGRMMxx PARMLIB OPTION TPRACF(N) command. You can set up the type of processing that you want by requesting automatic TPRACF(A) or predefined TPRACF(P) RACF profiles.

You can also use the DFSMSrmm EDGRMMxx PARMLIB VLPOOL RACF command to provide control of RACF at the pool level.

13.2.3 DFSMSrmm automatic tape security support

DFSMSrmm automatic tape security processing assumes that when a tape is mounted for output as a scratch tape, it is not RACF protected. During the open processing for the tape data set, either DFSMSdfp or your installation exits cause some RACF profiles to be created. DFSMSrmm predefined processing also ensures that when TAPEVOL and TAPEDSN are active, RACF predefined TAPEVOL profiles with an empty TVTOC are created for scratch volumes.

To enforce a DFSMSrmm standard that all tapes are RACF protected when the data set is closed, DFSMSrmm checks that a RACF security profile exists for the volume. If a RACF security profile does not exist, DFSMSrmm creates one and places the owner of the tape in the access list for the TAPEVOL profile with ALTER authority. The process helps to ensure that your current tape security mechanism continues to work as long as it is based on TAPEVOL profiles.

13.2.4 DFSMSrmm OPTION command operand TPRACF

This specifies the type of RACF tape support that DFSMSrmm provides. Use this operand when you want DFSMSrmm to maintain the security profiles that protect tape volumes. RACF tape support that you define can be overridden by RACF tape support that you define with the VLPOOL command described in 13.2.5, “DFSMSrmm VLPOOL command operand RACF” on page 486.

The TPRACF(NO) option is assumed for any volume serial number that contains special characters. To protect tape volumes that use special characters in the volume serial number, use RACF generic TAPEVOL profiles that are outside of DFSMSrmm control. DFSMSrmm honors all other TPRACF options for volume serial numbers that are alphanumeric, including national characters. You can also use RACF generic data set profiles to protect data that is created on the tape volume. DFSMSrmm honors the TPRACF setting when running in all modes.

If you set TPRACF(PREDEFINED) or TPRACF(AUTOMATIC), DFSMSrmm ensures that all non-scratch tapes are protected by a discrete RACF TAPEVOL profile by checking that a RACF profile exists, whenever a data set is written on a tape. If a profile does not exist, DFSMSrmm creates one. Therefore, you do not need to use RACF installation exits to set the JCL PROTECT=YES option or specify PROTECT=YES in your JCL. You can use generic data set profiles for all tape data sets without changes to JCL or installation procedures, if you used the VLPOOL command with the RACF(Y) operand, because DFSMSrmm creates a TVTOC when you use the RACF TAPEDSN option.

For both TPRACF(PREDEFINED) and TPRACF(AUTOMATIC), DFSMSrmm ensures TAPEVOL profiles are always deleted during the recycling of scratch volumes, if they exist. When you use a volume for output and there is no RACF protection for the volume when you close the data set, DFSMSrmm creates a profile to protect the volume. The owner of the volume is given RACF access to the profile. If TAPEVOL and TAPEDSN are active, DFSMSrmm creates a TVTOC and adds the first file data set entry. The default is TPRACF(N).

For more information about RACF processing and security options, see the chapter titled “Authorizing DFSMSrmm Users and Ensuring Security” in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

13.2.5 DFSMSrmm VLPOOL command operand RACF

You can specify the type of RACF tape support that DFSMSrmm provides for volumes in the pool that you are defining. When you define RACF tape support for volumes in a pool, you must look at the RACF tape support that you defined for your installation with the OPTION TPRACF command described in the chapter “Defining System Options: OPTION,” in the *DFSMSrmm Implementation and Customization Guide,* SG26-7405.

Specify `Y` if you want DFSMSrmm to create RACF tape profiles for the volumes in the pool. Ensure that OPTION TPRACF(AUTOMATIC or PREDEFINED) is specified. If OPTION TPRACF(NO) is specified, DFSMSrmm does not create the RACF tape profiles. Only specify `Y` for tape pools because TAPEVOL profiles are not required for optical volumes.

Specify `N` if you do not want DFSMSrmm to create RACF tape profiles for the volumes in the pool. If you specify `N`, DFSMSrmm plays no part in creating or deleting RACF tape profiles, regardless of the system-wide option TPRACF.

The default is RACF(N).

13.2.6 Running DFSMShsm and DFSMSrmm

How you use the DFSMShsm SETSYS option TAPESECURITY influences the TPRACF value in the DFSMSrmm PARMLIB member EDGRMMxx. DFSMShsm uses RACF TAPEVOL profiles to secure its volumes. You can combine the DFSMShsm method for securing volumes with DFSMSrmm RACF options as described in the chapter “Defining System Options: OPTION” in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405, to ensure complete security for your tape data.

Select the appropriate combinations for your environment from the following options:

- Use RACF with TAPEVOL and TAPEDSN.
- Use DFSMShsm TAPE SECURITY with RACF or EXPIRATIONINCLUDE.
- Use DFSMSrmm TPRACF with A, P, or N and VLPOOL RACF with Y or N.

Suggestions:

- Run DFSMShsm with scratch tapes that DFSMSrmm manages so that you can have a single scratch pool for all users of tape and gain any available benefits from pre-mounting scratch tapes in cartridge loaders.
- Use RACF tape security to ensure data security on your system.
- Set the following suggested DFSMShsm parameters:

```
SETSYS EXITOFF(ARCTVEXT)              -
SELECTVOLUME(SCRATCH)                 -
TAPEDELETION(SCRATCH)                 -
TAPESECURITY(RACF EXPIRATION)         -
PARTIALTAPE(MARKFULL)
```

  Specify EXITOFF(ARCTVEXT) if DFSMSrmm is your only tape management product because DFSMShsm always calls the DFSMSrmm programming interface EDGTVEXT if you have installed DFSMSrmm Release 1.4 or later.

13.2.7 Defining RACF profiles

Example 13-1 on page 487 shows a sample batch job to protect tape volumes and tape data sets using RACF RDEFINE, RALTER, and ADDSD commands.

Example 13-1 RACF commands for TAPEVOL and TAPEDSN implementation

```
/* ********************************************************** */
/* RACF cmds for DFSMSrmm TAPEVOL and/or TAPEDSN implementation */
/*                                                            */
/* ********************************************************** */
RDEFINE TAPEVOL HSMHSM OWNER(DFHSM) +
    UACC(NONE) AUDIT(ALL(READ))                      /* VOL-SET */
    PERMIT HSMHSM CLASS(TAPEVOL) ID(TRESTER) DELETE
RALTER TAPEVOL SC0239 TVTOC                           /* MASTER */
ADDSD 'BSYPPS.OPD.STAIR.TSTAIR.E9201' +
    FILESEQ(1) SETONLY TAPE UNIT(TAPE) VOL(SC0239)
ADDSD 'BSYPPS.OPD.STAIR.DSTAIR.E9201' +
    FILESEQ(2) SETONLY TAPE UNIT(TAPE) VOL(SC0239)
ADDSD 'BSYPPS.OPD.STAIR.DATASETS.E9201' +
    FILESEQ(3) SETONLY TAPE UNIT(TAPE) VOL(SC0239)
PERMIT SC0239 CLASS(TAPEVOL) ID(TRESTER) DELETE
RALTER TAPEVOL HSMHSM ADDVOL(SC0648)                    /* HBAC */
RALTER TAPEVOL SC0010 TVTOC                          /* MASTER */
ADDSD 'SCHLUM.RMMDEMO.FILE2.VOL12' +
    FILESEQ(1) SETONLY TAPE UNIT(TAPE) VOL(SC0010)
PERMIT SC0239 CLASS(TAPEVOL) ID(TRESTER) DELETE
```

The following definitions relate to Example 13-1:

RDEFINE TAPEVOL HSMHSM
DFSMShsm protects tapes with RACF by adding them to a RACF tape-volume set. All tapes in the DFSMShsm RACF tape-volume set share the same access list and auditing controls.

RALTER TAPEVOL SC0239 TVTOC
Create a TVTOC for an existing TAPEVOL profile.

ADDSD 'BSYPPS.OPD.STAIR.TSTAIR.E9201
Add tape data set profiles to the TVTOC for volumes with the DFSMSrmm status of MASTER.

PERMIT SC0239 CLASS(TAPEVOL) ID(TRESTER) DELETE
Remove the command executor from the access list.

13.2.8 Final RACF activities

After you define the DFSMSrmm RACF function groups, data set profiles, and resources in the RACF FACILITY class, and assign DFSMSrmm a user ID, you must refresh the RACF in-storage profiles.

Example 13-2 shows a batch job to REFRESH the RACF in-storage profiles.

Example 13-2 JCL to refresh RACF in-storage-profiles

```
//* ******************************************************************* *
//*  Caution
//*  =======
//*    To execute this job you need the RACF "SPECIAL" attribute
//* ******************************************************************* *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
```

```
//SYSTSIN  DD   *
 /* ************************************************************** */
 /*   FINAL ACTIVITIES                                             */
 /* ************************************************************** */
   SETROPTS GENERIC(FACILITY) REFRESH
   SETROPTS RACLIST(FACILITY) REFRESH
   SETROPTS GENERIC(DATASET) REFRESH
   SETROPTS GENERIC(STARTED) REFRESH
   SETROPTS LIST
//
```

13.3 Implementation SAF tape security

The implementation of the new tape data set authorization is described. The new implementation of tape data set authorization using the new DEVSUP settings to protect data sets on tape using the DATASET class is described.

Note: Before you start this new tape data set authorization implementation, search your DFSMSrmm control data set and check that for all data sets residing on tape, at a minimum, the high-level qualifier is defined to RACF.

Protect the use of the MVS SET DEVSUP command so that only a few people have access to this resource and are able to modify your tape data set authorization settings. Example 13-3 shows how you can protect the use of the MVS DEVSUP command.

Example 13-3 Protect MVS SET DEVSUP command

```
//* ****************************************************************
//* * TESTING Tape Data Set Authorization                           *
//* ****************************************************************
//DELETE   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEF OPERCMDS MVS.SET.DEVSUP UACC(NONE)
 PE MVS.SET.DEVSUP CL(OPERCMDS) ID(MHLRES5 MHLRES2) ACC(NONE)
 SETR REFRESH RACLIST(OPERCMDS)
/*
```

Note: Refresh the RACF OPERCMDS because OPERCMDS is in the RACLIST class.

Use the MVS SET DEVSUP command in Example 13-4 to test your RACF protection of the MVS SET DEVSUP command.

Example 13-4 Unauthorized use of set DEVSUP

```
/T DEVSUP=18
```

This MVS SET COMMAND is not working because the user MHLRES5 is not permitted to use this command. Figure 13-4 on page 489 shows the RACF error messages you receive.

```
RESPONSE=SC64
  ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
    MVS.SET.DEVSUP CL(OPERCMDS)
    INSUFFICIENT ACCESS AUTHORITY
    ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
```

Figure 13-4 RACF error messages

13.3.1 Check all high-level qualifiers on tape

Check that each high-level qualifier on volumes in DFSMSrmm status MASTER or USER is defined to RACF as a user or group. You can create different reports to check the high-level qualifiers that are currently used:

- Only on volumes in status SCRATCH
- Only on volumes with a status other than SCRATCH
- Using all volumes in any volume status

13.3.2 Update the DEVSUPxx parmlib member

Update the DEVSUPxx member in your parmlib to specify your installation default for device support options. DEVSUPxx is processed during the nucleus initialization program (NIP) phase of the IPL. After the IPL, you can use the system command SET DEVSUP=XX to activate the DEVSUP changes.

To enable the new tape data set protection, add the following parameters to the DEVSUPxx parmlib member:

TAPEAUTHDSN This enables tape authorization checks in the DATASET class.

TAPEAUTHF1 This enables additional tape authorization checks in the DATASET class for existing files on the same tape volume when any other file on the tape volume is opened.

TAPEAUTHRC4 Use this keyword to control PROTECTALL processing for tape data sets. This applies to the results of RACROUTE processing when both TAPEAUTHDSN=YES and TAPEAUTHF1=YES are specified.

TAPEAUTHRC8 Use this keyword as an aid to the implementation of TAPEAUTHDSN and TAPEAUTHF1.

It provides a managed and controlled implementation of tape authorization checks in the DATASET class, and it applies only to the results of TAPEAUTHDSN=YES and TAPEAUTHF1=YES processing.

Note: While TAPEAUTHRC4=FAIL and TAPEAUTHRC8=FAIL are specified to implement the z/OS DFSMS V1.8 tape data set security enhancement, the TAPEAUTHRC4=ALLOW and TAPEAUTHRC8=WARN options are provided to ease the migration to the new tape security implementation. It is suggested that you initially specify TAPEAUTHRC4=ALLOW and TAPEAUTHRC8=WARN.

Figure 13-5 on page 490 shows a sample DEVSUPxx PARMLIB member, including the default for compaction for new data sets written to 3480, 3490, or 3590 tape subsystems. The category codes used for this system for MEDIA1, MEDIA2, MEDIA3, and MEDIA5 tapes in scratch status are specified. The PRIVATE category for tapes in private status and the ERROR category for all volumes for which OAM detects any error are also specified.

This member also includes the final settings for the new tape data set authorization implementation after you have successfully tested this new function.

```
COMPACT = YES,                    /* INSTALLATION DEFAULT FOR IDRC */
MEDIA1 = 0021,
MEDIA2 = 0022,
MEDIA3 = 0023,
MEDIA5 = 0025,
ERROR = 002E,
PRIVATE = 002F,
TAPEAUTHDSN = YES,
TAPEAUTHF1 = YES,
TAPEAUTHRC4 = FAIL,
TAPEAUTHRC8 = FAIL
```

Figure 13-5 Sample DEVSUPxx parmlib member

The following definitions apply to Figure 13-5:

COMPACT Specifies the default for compaction for new data sets written to 3480, 3490, or 3590 tape subsystems.

MEDIAx Used to specify category codes for library partitioning.

ERROR Used to specify category codes for library partitioning.

PRIVATE Used to specify category codes for library partitioning.

TAPEAUTHDSN=YES
Enables tape authorization checks in the DATASET class but without DSTYPE=T. DSTYPE=T indicates to RACF that the check is for a data set on a tape volume and that special RACF tape data set and tape volume processing is to be performed. Without DSTYPE=T, RACF authorization checking considers only profiles in the DATASET class. The system uses the data set name specified in the allocation or JCL to check your authorization to read or write the specified file. In addition, the system determines the RACF erase-on-scratch setting from the RACF profile and passes it to your tape management system. Use this option only when you have a tape management system, such as DFSMSrmm, installed and actively checking that the 44-character data set name specified by the user matches the data set name on tape. Without a tape management system, tape data set open processing can only validate the last 17 characters of the data set name against the tape volume labels. When you request BLP and the mounted volume uses standard labels, OPEN issues the authorization check that the user is authorized to use BLP. This processing uses the existing ICHBLP resource in the RACF FACILITY class. When you specify TAPEAUTHDSN=YES only, it replaces the check that RACF makes as part of tape volume authorization checking.

TAPEAUTHF1=YES Enables additional tape authorization checks in the DATASET class for existing files on the same tape volume when any other file on the tape volume is opened. This function depends on the tape management system returning the 44-character data set name and data set sequence number to OPEN/EOV through the IFGTEP during the Volume Mount exit Volume Security function. If no data set name is returned by the tape management system, processing occurs as though this keyword was not specified. Although it is intended to enable an additional authorization check for the first data set when any other data set on the tape volume is opened, the implementation allows your tape management system to request one or more additional authorization checks when any data set on a tape volume is opened. Each additional data set name and data set sequence number that are returned result in an additional RACROUTE. Do not use this function unless you have a tape management system and it can return a data set name and data set sequence number. A data set sequence number is the label number normally specified in the JCL LABEL keyword and stored in the catalog. When TAPEAUTHDSN=YES is in use, any additional RACROUTE matches that issued for TAPEAUTHDSN, except for the data set name and data set sequence number. Otherwise, TAPEAUTHF1 uses a RACROUTE that matches that used for SETROPTS TAPEDSN. When neither TAPEAUTHDSN nor SETROPTS TAPEDSN is in use, TAPEAUTHF1 support is not provided.

TAPEAUTHRC4 Denies the access of data sets that are not protected by a security profile.

TAPEAUTHRC8 Denies the access of data sets that typically cannot be accessed.

Use the MVS SET DEVSUP command to implement the new tape data set security settings. Figure 13-6 shows the successful result of the command.

```
T DEVSUP=18
IEE252I MEMBER DEVSUP18 FOUND IN SYS1.PARMLIB
IEE536I DEVSUP   VALUE 18 NOW IN EFFECT
IEA253I DEVSUP  3480X RECORDING MODE DEFAULT IS COMPACTION.
IEA253I DEVSUP  ISO/ANSI TAPE LABEL VERSION DEFAULT IS V3
IEA253I DEVSUP  TAPE OUTPUT DEFAULT BLOCK SIZE LIMIT IS 32760
IEA253I DEVSUP  COPYSDB DEFAULT IS INPUT
IEA253I DEVSUP  TAPEAUTHDSN: YES
IEA253I DEVSUP  TAPEAUTHF1: YES
IEA253I DEVSUP  TAPEAUTHRC4: FAIL
IEA253I DEVSUP  TAPEAUTHRC8: FAIL
```

Figure 13-6 Result of the SET DEVSUP command

Restriction: Only the new tape security parameters are updated by using the SET DEVSUP command. All existing parameters are not updated and cannot be changed without an IPL.

13.4 Removing TAPEVOL and TAPEDSN processing

Instructions are provided for changing from using RACF TAPEVOL and TAPEDSN to the new tape data set authorization checking.

13.4.1 Check and modify your RACF settings

First, check your RACF settings using the RACF SETR LIST command in a TSO session as shown in Example 13-5.

Example 13-5 RACF SETR LIST command used in TSO

```
Command ===> SETR LIST
```

Example 13-6 shows the same command used in a batch job.

Example 13-6 RACF SETR LIST command used in a batch job

```
//SETRLIST EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR LIST
/*
```

The following examples show the output with the current RACF settings. Example 13-7 shows the RACF ATTRIBUTES, STATISTICS, and ACTIVE CLASSES.

Example 13-7 RACF ATTRIBUTES, STATISTICS, and ACTIVE CLASSES

```
ATTRIBUTES = INITSTATS WHEN(PROGRAM -- BASIC)
STATISTICS = DATASET DASDVOL GDASDVOL GTERMINL O2OMPE TAPEVOL TERMINAL
ACTIVE CLASSES = DATASET USER GROUP ACCTNUM ACICSPCT APPCLU APPCPORT APPCSERV
                 APPCTP APPL BCICSPCT CBIND CCICSCMD CDT CONSOLE CPSMOBJ
                 CPSMXMP CSFKEYS CSFSERV DASDVOL DCICSDCT DIGTCERT DIGTRING
                 DSNR ECICSDCT EJBROLE FACILITY FCICSFCT FIELD FSSEC GCICSTRN
                 GCPSMOBJ GCSFKEYS GDASDVOL GEJBROLE GMQADMIN GSDSF GXFACILI
                 HCICSFCT IBMOPC ILMADMIN JCICSJCT JESJOBS JESSPOOL KCICSJCT
                 KEYSMSTR LOGSTRM MCICSPPT MQADMIN NCICSPPT NETCMDS NETSPAN
                 NODES NODMBR OPERCMDS O2OMPE PCICSPSB PMBR PRINTSRV PROGRAM
                 PTKTDATA PTKTVAL QCICSPSB RACFVARS RACGLIST RRSFDATA
                 RVARSMBR SCICSTST SDSF SERVAUTH SERVER STARTED STORCLAS
                 SURROGAT SYSMVIEW TAPEVOL TCICSTRN TMEADMIN TSOAUTH TSOPROC
                 UCICSTST UNIXPRIV VCICSCMD VTAMAPPL WRITER XFACILIT
```

In Example 13-8, you can see the active RACF GENERIC PROFILE CLASSES.

Example 13-8 RACF GENERIC PROFILE CLASSES

```
GENERIC PROFILE CLASSES =  DATASET ACCTNUM ACICSPCT AIMS ALCSAUTH APPCLU
                           APPCPORT APPCSERV APPCSI APPCTP APPL CACHECLS
                           CBIND CCICSCMD CIMS CONSOLE CPSMOBJ CPSMXMP
                           CSFKEYS CSFSERV DASDVOL DBNFORM DCEUUIDS DCICSDCT
                           DEVICES DIGTCERT DIGTCRIT DIGTNMAP DIGTRING
                           DIRACC DIRAUTH DIRECTRY DIRSRCH DLFCLASS DSNADM
                           DSNR EJBROLE FACILITY FCICSFCT FIELD FILE FIMS
                           FSOBJ FSSEC GMBR IBMOPC ILMADMIN INFOMAN IPCOBJ
                           JAVA JCICSJCT JESINPUT JESJOBS JESSPOOL KEYSMSTR
```

```
                                LDAPBIND LFSCLASS LOGSTRM MCICSPPT MDSNBP MDSNCL
                                MDSNDB MDSNJR MDSNPK MDSNPN MDSNSC MDSNSG MDSNSM
                                MDSNSP MDSNTB MDSNTS MDSNUF MDSNUT MGMTCLAS
                                MQADMIN MQCHAN MQCMDS MQCONN MQNLIST MQPROC
                                MQQUEUE NDSLINK NETCMDS NETSPAN NODES NODMBR
                                NOTELINK NVASAPDT OIMS OPERCMDS O2OMPE PCICSPSB
                                PERFGRP PIMS PMBR PRINTSRV PROCACT PROCESS
                                PROPCNTL PSFMPL PTKTDATA PTKTVAL RACFVARS RACGLIST
                                RMTOPS RODMMGR ROLE RRSFDATA RVARSMBR SCDMBR
                                SCICSTST SDSF SECLMBR SERVAUTH SERVER SFSCMD
                                SIMS SMESSAGE SOMDOBJS STARTED STORCLAS SUBSYSNM
                                SURROGAT SYSMVIEW TAPEVOL TCICSTRN TEMPDSN
                                TERMINAL TIMS TMEADMIN TSOAUTH TSOPROC UNIXMAP
                                UNIXPRIV VMBATCH VMBR VMCMD VMMAC VMMDISK VMNODE
                                VMPOSIX VMRDR VMSEGMT VTAMAPPL VXMBR WRITER
                                XFACILIT
```

Example 13-9 shows the current active RACF GENERIC COMMAND CLASSES.

Example 13-9 RACF GENERIC COMMAND CLASSES

```
GENERIC COMMAND CLASSES =  DATASET ACCTNUM ACICSPCT AIMS ALCSAUTH APPCLU
                           APPCPORT APPCSERV APPCSI APPCTP APPL CACHECLS
                           CBIND CCICSCMD CIMS CONSOLE CPSMOBJ CPSMXMP
                           CSFKEYS CSFSERV DASDVOL DBNFORM DCEUUIDS DCICSDCT
                           DEVICES DIGTCERT DIGTCRIT DIGTNMAP DIGTRING
                           DIRACC DIRAUTH DIRECTRY DIRSRCH DLFCLASS DSNADM
                           DSNR EJBROLE FACILITY FCICSFCT FIELD FILE FIMS
                           FSOBJ FSSEC GMBR IBMOPC ILMADMIN INFOMAN IPCOBJ
                           JAVA JCICSJCT JESINPUT JESJOBS JESSPOOL KEYSMSTR
                           LDAPBIND LFSCLASS LOGSTRM MCICSPPT MDSNBP MDSNCL
                           MDSNDB MDSNJR MDSNPK MDSNPN MDSNSC MDSNSG MDSNSM
                           MDSNSP MDSNTB MDSNTS MDSNUF MDSNUT MGMTCLAS
                           MQADMIN MQCHAN MQCMDS MQCONN MQNLIST MQPROC
                           MQQUEUE NDSLINK NETCMDS NETSPAN NODES NODMBR
                           NOTELINK NVASAPDT OIMS OPERCMDS O2OMPE PCICSPSB
                           PERFGRP PIMS PMBR PRINTSRV PROCACT PROCESS
                           PROPCNTL PSFMPL PTKTDATA PTKTVAL RACFVARS RACGLIST
                           RMTOPS RODMMGR ROLE RRSFDATA RVARSMBR SCDMBR
                           SCICSTST SDSF SECLMBR SERVAUTH SERVER SFSCMD
                           SIMS SMESSAGE SOMDOBJS STARTED STORCLAS SUBSYSNM
                           SURROGAT SYSMVIEW TAPEVOL TCICSTRN TEMPDSN
                           TERMINAL TIMS TMEADMIN TSOAUTH TSOPROC UNIXMAP
                           UNIXPRIV VMBATCH VMBR VMCMD VMMAC VMMDISK VMNODE
                           VMPOSIX VMRDR VMSEGMT VTAMAPPL VXMBR WRITER
                           XFACILIT
```

Example 13-10 on page 494 shows the RACF GENLIST, GLOBAL CHECKING, and the settings of the RACLIST classes.

Example 13-10 RACF RACLIST CLASSES

```
GENLIST CLASSES =  NONE
GLOBAL CHECKING CLASSES =  O2OMPE
SETR RACLIST CLASSES =  ACCTNUM APPCPORT APPCSERV APPCTP APPL CBIND CDT
                        CSFKEYS CSFSERV DIGTCERT DIGTRING FACILITY FIELD
                        ILMADMIN JESSPOOL NETCMDS NODES OPERCMDS O2OMPE
                        PRINTSRV PTKTDATA PTKTVAL RACFVARS RRSFDATA SDSF
                        SERVAUTH SERVER STARTED SURROGAT SYSMVIEW TSOAUTH
                        TSOPROC UNIXPRIV VTAMAPPL WRITER XFACILIT
```

In Example 13-11, you can see that the tape data set protection is active (TAPEDSN) and that the high-level qualifier “PASSWORD” will be added to each single-level qualifier data set that is used on this system.

Example 13-11 Other RACF information

```
GLOBAL=YES RACLIST ONLY =  NONE
AUTOMATIC DATASET PROTECTION IS NOT IN EFFECT
ENHANCED GENERIC NAMING IS IN EFFECT
REAL DATA SET NAMES OPTION IS INACTIVE
JES-BATCHALLRACF OPTION IS INACTIVE
JES-XBMALLRACF OPTION IS INACTIVE
JES-EARLYVERIFY OPTION IS ACTIVE
PROTECT-ALL OPTION IS NOT IN EFFECT
TAPE DATA SET PROTECTION IS ACTIVE
SECURITY RETENTION PERIOD IN EFFECT IS  9999 DAYS.
ERASE-ON-SCRATCH IS INACTIVE
SINGLE LEVEL NAME PREFIX IS PASSWORD
LIST OF GROUPS ACCESS CHECKING IS ACTIVE.
INACTIVE USERIDS ARE NOT BEING AUTOMATICALLY REVOKED.
NO DATA SET MODELLING BEING DONE.
PASSWORD PROCESSING OPTIONS:
  PASSWORD CHANGE INTERVAL IS 180 DAYS.
  PASSWORD MINIMUM CHANGE INTERVAL IS   0 DAYS.
  MIXED CASE PASSWORD SUPPORT IS NOT IN EFFECT
  NO PASSWORD HISTORY BEING MAINTAINED.
  USERIDS NOT BEING AUTOMATICALLY REVOKED.
  NO PASSWORD EXPIRATION WARNING MESSAGES WILL BE ISSUED.
  NO INSTALLATION PASSWORD SYNTAX RULES ARE PRESENT.
DEFAULT RVARY PASSWORD IS IN EFFECT FOR THE SWITCH FUNCTION.
DEFAULT RVARY PASSWORD IS IN EFFECT FOR THE STATUS FUNCTION.
SECLABEL CONTROL IS NOT IN EFFECT
GENERIC OWNER ONLY IS IN EFFECT
COMPATIBILITY MODE IS NOT IN EFFECT
MULTI-LEVEL QUIET IS NOT IN EFFECT
MULTI-LEVEL STABLE IS NOT IN EFFECT
NO WRITE-DOWN IS NOT IN EFFECT
MULTI-LEVEL ACTIVE IS NOT IN EFFECT
CATALOGUED DATA SETS ONLY, IS NOT IN EFFECT
USER-ID FOR JES NJEUSERID IS : ????????
USER-ID FOR JES UNDEFINEDUSER IS : ++++++++
PARTNER LU-VERIFICATION SESSIONKEY INTERVAL DEFAULT IS "NEVER EXPIRES".
ADDCREATOR IS NOT IN EFFECT
KERBLVL =     1
MULTI-LEVEL FILE SYSTEM IS NOT IN EFFECT
MULTI-LEVEL INTERPROCESS COMMUNICATIONS IS NOT IN EFFECT
MULTI-LEVEL NAME HIDING IS IN EFFECT
SECURITY LABEL BY SYSTEM IS NOT IN EFFECT
PRIMARY LANGUAGE DEFAULT : ENU
```

```
SECONDARY LANGUAGE DEFAULT : ENU
```

In Example 13-11 on page 494, you can see that the RACF TAPEVOL class is active and TAPE DATA SET PROTECTION is set, so we deactivate both functions by using the RACF commands as shown in Example 13-12.

Example 13-12 Deactivate the RACF TAPEVOL class and TAPE DATA SET PROTECTION

```
//DELETE   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETROPTS NOCLASSACT(TAPEVOL)
 SETROPTS NOTAPEDSN
 SETR REFRESH GENERIC(TAPEVOL)
/*
```

13.4.2 Check and modify your DFSMShsm settings

Check your DFSMShsm settings and switch off the TAPESECURITY if this option is set. Figure 13-7 shows a sample ARCCMDxx parmlib member where you can see that the SETSYS TAPESECURITY(RACF) option is used.

The use of TAPESECURITY(RACF) means that DFSMShsm protects each backup, migration, and dump tape with RACF. DFSMShsm also protects alternate backup and migration tapes generated as a result of TAPECOPY processing. The RACF subparameter does not support backup or migration of password-protected data sets.

Note: The RACF option of the SETSYS TAPESECURITY command directs DFSMShsm to automatically add RACF protection to scratch tapes.

```
/*****************************************************************/
/*                DFSMSHSM RACF SPECIFICATIONS                   */
/*****************************************************************/
/*                                                               */
SETSYS                      /* DO NOT PUT RACF-INDICATION        */ -
  NORACFIND                 /* ON BACKUP AND MIGRATION           */
                            /* COPIES OF DATA SETS.              */
SETSYS                      /* USE RACF TO PROVIDE TAPE SECURITY */ -
  TAPESECURITY(RACF EXPIRATIONINCLUDE)        /* USE 99365 EXPDT */
SETSYS                      /* DO NOT ALLOW ERASE-ON-SCRATCH     */ -
  NOERASEONSCRATCH          /* ON ANY DFSMSHSM BACKUP            */
                            /* VERSIONS AND MIGRATION COPIES     */
SETSYS                      /* BACKUP DISCRETE RACF PROFILES     */ -
  PROFILEBACKUP
...
```

Figure 13-7 Sample DFSMShsm PARMLIB member

If the TAPESECURITY is set to RACF, change your ARCCMDxx parmlib member as shown in Figure 13-8 on page 496 to remove the RACF setting. Restart your DFSMShsm so that the new setting is used.

```
/*****************************************************************/
/*                DFSMSHSM RACF SPECIFICATIONS                   */
/*****************************************************************/
SETSYS                      /* DO NOT PUT RACF-INDICATION        */ -
  NORACFIND                 /* ON BACKUP AND MIGRATION           */
                            /* COPIES OF DATA SETS.              */
SETSYS                      /* USE 99365 TO WIRTE  AN EXPDT      */ -
  TAPESECURITY(EXPIRATIONINCLUDE)      /*  TO EACH TAPE         */
SETSYS                      /* DO NOT ALLOW ERASE-ON-SCRATCH     */ -
  NOERASEONSCRATCH          /* ON ANY DFSMSHSM BACKUP            */
                            /* VERSIONS AND MIGRATION COPIES     */
SETSYS                      /* BACKUP DISCRETE RACF PROFILES     */ -
  PROFILEBACKUP
...
```

Figure 13-8 Sample DFSMShsm PARMLIB member with RACF setting

Note: You must restart your DFSMShsm because there is no SETSYS command available to switch off the RACF setting.

13.4.3 Clean up your TAPEVOL profiles by using DFSMSrmm settings

Modify your DFSMSrmm settings in EDGRMMxx if you specified that DFSMSrmm maintains the security profiles that protect tape volumes specified in the TPRACF OPTION operand.

TPRACF

Check the type of RACF tape support that you have selected. If you specified TPRACF(AUTOMATIC) or TPRACF(PREDEFINED), change this setting to one of the following settings:

TPRACF(CLEANUP) DFSMSrmm ensures that TAPEVOL profiles and discrete tape DATASET profiles are deleted during recycling of scratch volumes and existing TAPEVOL profiles are deleted when volumes are deleted from the DFSMSrmm control data set (CDS). When you use this option, DFSMSrmm never creates any RACF profiles for you. This processing is only provided for VLPOOLs with RACF(Y). TPRACF(CLEANUP) is intended to be used when you are changing how tape data sets are protected. For example, if you no longer want to use TAPEVOL profiles and are enabling the use of DATASET profiles, TPRACF(CLEANUP) can be used for this situation. When you specify TPRACF(CLEANUP), DFSMSrmm deletes RACF tape profiles for any volumes in your installation based on the following VLPOOL values:

VLPOOL RACF(N) DFSMSrmm does no processing of tape profiles for volumes in the pool at any time.

VLPOOL RACF(Y), RACF
Tape profiles are deleted when RMM CHANGEVOLUME or DELETEVOLUME subcommands are issued. DFSMSrmm deletes TAPEVOL and discrete tape data set profiles during the recycling of scratch tapes if the profiles exist.

TPRACF(NONE) DFSMSrmm will not manipulate a TAPEVOL profile in any circumstance.

Example 13-13 shows how to update to your EDGRMMxx parmlib member OPTION command if you want DFSMSrmm to clean up your security TAPEVOL profiles.

Example 13-13 Update the OPTION command

```
OPTION    OPMODE(P)                  /* Protect Mode            */ -
          ...
          CATSYSID(*)                /* all catalogs shared     */ -
          CDSID(PROD)                /* control data set id     */ -
          COMMANDAUTH(OWNER)         /* type of authorization   */ -
          SMFAUD(YES)                /* SMF audit records       */ -
          SMFSEC(YES)                /* SMF security records    */ -
          ...                                                      -
          TPRACF(CLEANUP)            /* RACF tape support       */ -
          TVEXTPURGE(EXPIRE)         /* set an expiration date  */ -
          ....
```

Example 13-14 shows the correct setting of the VLPOOL command in the EDGRMMnn parmlib member.

Example 13-14 Update the VLPOOL command

```
VLPOOL    PREFIX(TST*)               /* volume prefix             */ -
          AUTOSCRATCH(YES)           /* return to scratch         */ -
          TYPE(S)                    /* satisfy scratch requests  */ -
          RACF(Y)                    /* support TAPEVOL/TAPEDSN   */ -
          MASTEROVERWRITE(USER)      /* allow to overwrite DSNs   */ -
          MEDIANAME(3490)            /* media name used           */ -
          EXPDTCHECK(N)              /* do not check VOL1 expdt   */ -
          DESCRIPTION('LOGISCHE VTS VOLUMES')
VLPOOL    PREFIX(E*)                 /* volume prefix             */ -
          AUTOSCRATCH(YES)           /* return to scratch         */ -
          TYPE(S)                    /* satisfy scratch requests  */ -
          RACF(Y)                    /* support TAPEVOL/TAPEDSN   */ -
          MASTEROVERWRITE(USER)      /* allow to overwrite DSNs   */ -
          MEDIANAME(3490)            /* media name used           */ -
          EXPDTCHECK(N)              /* do not check VOL1 expdt   */ -
          DESCRIPTION('Enhanced 3590 cartridges')
VLPOOL    PREFIX(M*)                 /* volume prefix             */ -
          AUTOSCRATCH(YES)           /* return to scratch         */ -
          TYPE(S)                    /* satisfy scratch requests  */ -
          RACF(Y)                    /* support TAPEVOL/TAPEDSN   */ -
          MASTEROVERWRITE(USER)      /* allow to overwrite DSNs   */ -
          MEDIANAME(3490)            /* media name used           */ -
          EXPDTCHECK(N)              /* do not check VOL1 expdt   */ -
          DESCRIPTION('MEDIA5 cartridges')
```

The following definition relates to Example 13-14:

RACF(Y) Specifies that you want DFSMSrmm to delete RACF tape profiles for the volumes in the pool

13.4.4 Clean up your TAPEVOL profiles by using commands

If you want to clean up your security server directly, you can use the TSO RMM SEACHVOLUME subcommand with the CLIST operand to create RACF delete TAPEVOL commands. After you check the commands, you can execute them to delete all your existing TAPEVOL profiles directly. Example 13-15 on page 498 shows sample JCL to create the RDELETE statements.

Example 13-15 Create RACF RDELETE commands

```
//CLEANUP  DD   PGM=IDCAMS
//SYSPRINT DD   DUMMY
//SYSIN    DD   *
  DELETE RMM.DELETE.RACF.TAPEVOL.TST NONVSAM PURGE
/*
//LCLSVSM  EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSTSPRT DD   SYSOUT=*
//RMMCLIST DD   DISP=(,CATLG),DSN=RMM.DELETE.RACF.TAPEVOL.TST,
//         SPACE=(TRK,(45,45),RLSE),LRECL=80,RECFM=FB,
//         UNIT=SYSDA
//SYSTSIN  DD   *
 RMM SV VOLUME(TST*) LIMIT(*) OWNER(*) -
     CLIST('RDELETE TAPEVOL ','')
/*
```

The following definitions relate to Example 13-15:

RMMCLIST DD The RMM TSO SEARCH subcommands with the CLIST operand are writing to this data set to store the commands.

CLIST(prefix_string,suffix_string)
Specifies a CLIST to create a data set of executable commands or to prepare an import list. You can edit the data set to remove any volumes that you do not want in the list. Then, you can run the CLIST at your convenience. DFSMSrmm returns the volume serial number for each record if you do not specify (prefix_string and suffix_string). When the volume serial number contains special characters, the value is returned within quotation marks. You can add RMM TSO subcommands and operands to the records in the CLIST data set by specifying (prefix_string and suffix_string). These text strings cannot exceed 255 characters. Separate the prefix_string and suffix_string by using a blank or a comma between the text strings. Insert blanks in the prefix and suffix values to prevent DFSMSrmm from concatenating the strings with the data that DFSMSrmm returns. To enter a null prefix_string, add a pair of separator characters such as ” to the text string (for example, CLIST(”,’ suffix_string’)).

Figure 13-9 shows how the RACF commands are created using the job in Example 13-15.

```
 RDELETE TAPEVOL TST000
RDELETE TAPEVOL TST001
RDELETE TAPEVOL TST002
RDELETE TAPEVOL TST003
RDELETE TAPEVOL TST004
RDELETE TAPEVOL TST005
RDELETE TAPEVOL TST006
RDELETE TAPEVOL TST007
....
```

Figure 13-9 Sample RACF RDELETE commands

After you have checked the commands, you can execute them using the JCL shown in Example 13-16 on page 499.

Example 13-16 Execute RACF RDELETE commands

```
//EXECUTE  EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 EX 'RMM.DELETE.RACF.TAPEVOL.TST'
/*
```

13.5 Error messages

Until the TAPEVOL profiles are deleted, you might see some security violations. Check your system log from time to time for messages.

Figure 13-10 shows message IEC150I, which is generated if you do not have the correct access to create a given data set.

```
JOB22272  IEC150I 913-60,IFG0196T,GENER99,STEP01,SYSUT2,0B91,TST026,DTAUS
JOB22272  IEA995I SYMPTOM DUMP OUTPUT  928
          SYSTEM COMPLETION CODE=913  REASON CODE=00000060
           TIME=20.17.17  SEQ=02226  CPU=0000  ASID=002A
           PSW AT TIME OF ERROR  075C1000   80C49CBE  ILC 2  INTC 0D
             NO ACTIVE MODULE FOUND
             NAME=UNKNOWN
             DATA AT PSW  00C49CB8 - 41003B7A  0A0D41F0  38BE56F0
             GR 0: 00000000_00C49F84   1: 00000000_A4913000
                2: 00000000_000097F4   3: 00000000_00C4940A
                4: 00000000_007DD1F8   5: 00000000_007DD58C
                6: 00000000_007DD534   7: 00000000_007DD58C
                8: 00000000_007DD554   9: 00000000_007C71A8
                A: 00000000_007DBCE0   B: 00000000_00C4CB04
                C: 00000000_80C4CBE4   D: 00000000_007DD4B8
                E: 00000000_80C49542   F: 00000010_00000060
           END OF SYMPTOM DUMP
JOB22272  IEF450I GENER99 STEP01 - ABEND=S913 U0000 REASON=00000060  929
```

Figure 13-10 IEC150I Message

The IEC150I format is shown:

IEC150I 913-*rc,mod,jjj,sss,ddname[-#],dev,ser,dsname(member)*

The following definitions relate to the IEC150I message:

rc Associates this message with system completion code 913 and with the return code.

jjj The job name.

sss The step name.

ddname[-#] DDname (followed by a concatenation number if it is part of a concatenation and not the first DD statement in the concatenation).

dev The device number.

ser The volume serial number.

mod The name of the module in which the error occurred.

***dsname*(*member*)** The data set name. Member name if specified. The explanation for the hex return code is for RACF errors, see message IDC3009I for the return code.

Severity Information.

Explanation The error occurred during one of the following situations. The processing of an OPEN macro instruction or during end of volume for a password-protected data set after the operator attempted to enter a password in response to message IEC301A. Or, the processing of an OPEN macro instruction involving a checkpoint data set. A Virtual Storage Access Method (VSAM) data set is being opened with a data control block (DCB) instead of an access method control block (ACB).

The explanation for the hex return code is for RACF errors, see message IDC3009I for the return code.

Return code	Explanation
60	One of the following situations occurred. The user is not authorized to define a data set with the specified name. The specified data set name and file sequence indicator do not match the corresponding names in the TVTOC. The user is not authorized to access this data set.

Important: If the volume gets the volume error status of "`SECURITY CONFLICT`" in the VOLCAT, it is protected for future use until you reset this situation if the volume is storage management subsystem (SMS)-managed:

```
 LINE       VOLUME USE      VOLUME              CHECKPT  LIBRARY   STORAGE
 OPERATOR   SERIAL ATTR     ERROR STATUS        VOLUME   NAME      GRP NAME
---(1)----  -(2)-- --(3)--  -------(4)--------  --(5)--  --(6)---  --(7)---
            TST020 SCRATCH  SECURITY CONFLICT   NO       LIB1      *SCRTCH*
----------  ------ -----------  BOTTOM  OF  DATA  -----------  ------  ----
```

Figure 13-11 on page 501 shows an ICH408I message that is shown for tapes. You currently get this message for data sets residing on DASD volumes if you do not have the correct access to the specified security resource in the RACF FACILITY class.

```
JOB22265  IEF233A M 0B91,PRIVAT,SL,GENER99,STEP01,MHLRES1.TESTSTAC.TESTYF01
JOB22265  ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)  781
            MHLRES1.TESTSTAC.TESTYF01 CL(DATASET ) VOL(TST020)
            INSUFFICIENT ACCESS AUTHORITY
            FROM MHLRES1.** (G)
            ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
JOB22265  IEC518I SOFTWARE ERRSTAT: RACFPROT 0B91,TST020,SL,GENER99,STEP01
JOB22265  IEC502E RK 0B91,TST020,SL,GENER99,STEP01
JOB22265  IEC150I 913-38,IFG0194F,GENER99,STEP01,SYSUT2,0B91,,MHLRES1.TESTSTAC.T
JOB22265  IEA995I SYMPTOM DUMP OUTPUT  785
          SYSTEM COMPLETION CODE=913  REASON CODE=00000038
           TIME=19.54.16  SEQ=02221  CPU=0000  ASID=002A
JOB22265  IEF450I GENER99 STEP01 - ABEND=S913 U0000 REASON=00000038  786
                  TIME=19.54.16
JOB22265  -                                        --TIMINGS (MINS.)--
JOB22265  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV
JOB22265  -GENER99           STEP01   *S913     51    .00    .00    .56    362
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Figure 13-11 Normal ICH408I security violation

Note: If the volume is an SMS-managed volume, the status of the volume is not changed in the VOLCAT and the volume can be continually accessed.

13.6 Testing various security settings

The following jobs give you an overview of how various tape security settings work and the error messages you get if your job abends. Table 13-2 shows the different settings we tested.

Table 13-2 Different RACF, DEVSUP, and DFSMSrmm settings

Test case/ success	RACF settings					TAPEAUTH settings in the DEVSUPnn				DFSMSrmm [2]	
	TAPEVOL	TAPEDSN	THM00n[4]	MHLRES1.**	MHLRES7.**	DSN	F1	RC4	RC8	TPRACF	RACF
All test cases between 1 and 19 are without the use of the new TAPEAUTH settings in the DEVSUPnn member. Create two new data sets on a tape volume, but at this time there are no DEVSUP settings.											
1 ·	ACTIVE	ACTIVE	N/A [1]	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
Read the previously created data sets with different RACF settings and different access to the DATASET and TAPEVOL profile											
2 ·	ACTIVE	ACTIVE	ALTER	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
3 ·	ACTIVE	ACTIVE	ALTER	NONE	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
4 ·	ACTIVE	ACTIVE	NONE	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
5 ✍	ACTIVE	ACTIVE	NONE	NONE	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
6 ·	N/A [1]	ACTIVE	N/A [1]	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
7 ✍	N/A [1]	ACTIVE	N/A [1]	NONE	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
8 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
9 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	NONE	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
Create an additional file to the previously created two tape data sets with different RACF settings.											
10 ·	ACTIVE	ACTIVE	ALTER	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y

Test case/ success	RACF settings					TAPEAUTH settings in the DEVSUPnn				DFSMSrmm [2]	
	TAPEVOL	TAPEDSN	THM00n[4]	MHLRES1.**	MHLRES7.**	DSN	F1	RC4	RC8	TPRACF	RACF
11 ✍	ACTIVE	ACTIVE	ALTER	ALTER	NONE	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
12 ·	ACTIVE	N/A [1]	ALTER	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
13 ·	ACTIVE	N/A [1]	ALTER	ALTER	NONE	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
14 ·	N/A [1]	ACTIVE	ALTER	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
15 ✍	N/A [1]	ACTIVE	ALTER	ALTER	NONE	N/A [1]	N/A [1]	N/A [1]	N/A [1]	A [3]	Y
Read the previously created data sets, but at this time the DFSMSrmm RACF support is switched off.											
16 ·	ACTIVE	ACTIVE	ALTER	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	**N**	**N**
17 ·	ACTIVE	ACTIVE	ALTER	ALTER	NONE	N/A [1]	N/A [1]	N/A [1]	N/A [1]	**N**	**N**
18 ·	N/A [1]	ACTIVE	N/A [1]	ALTER	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	**N**	**N**
19 ✍	N/A [1]	ACTIVE	N/A [1]	NONE	ALTER	N/A [1]	N/A [1]	N/A [1]	N/A [1]	**N**	**N**
All of the following test cases are now tested with the new TAPEAUTH settings in the DEVSUPnn member and the RACF CLASS TAPEVOL is not active and option TAPEDSN is inactive. Create two new data sets on a tape volume, but at this time RACF TAPEVOL and TAPEDSN are inactive and the DFSMSrmm RACF support is switched off.											
20 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	YES	YES	FAIL	Fail	N	N
Read the previously created data sets with different RACF access to MHLRES1.** and MHLRES7.**											
21 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	YES	YES	FAIL	FAIL	N	N
22 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	ALTER	YES	YES	FAIL	FAIL	N	N
23 ✍	N/A [1]	N/A [1]	N/A [1]	ALTER	NONE	YES	YES	FAIL	FAIL	N	N
24 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	NONE	YES	YES	FAIL	FAIL	N	N
Create an additional file MHLRES7.RACF.TEST2.FILEn with different RACF access to MHLRES1.** and MHLRES5.**											
25 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	YES	YES	FAIL	FAIL	N	N
26 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	ALTER	YES	YES	FAIL	FAIL	N	N
27 ✍	N/A [1]	N/A [1]	N/A [1]	ALTER	NONE	YES	YES	FAIL	FAIL	N	N
28 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	NONE	YES	YES	FAIL	FAIL	N	N
Read the previously created data sets with different RACF access to MHLRES1.** and MHLRES7.**											
29 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	YES	**NO**	FAIL	FAIL	N	N
30 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	ALTER	YES	**NO**	FAIL	FAIL	N	N
31 ✍	N/A [1]	N/A [1]	N/A [1]	ALTER	NONE	YES	**NO**	FAIL	FAIL	N	N
32 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	NONE	YES	**NO**	FAIL	FAIL	N	N
Create an additional file MHLRES1.RACF.TEST2.FILEn with different RACF access to MHLRES1.** and MHLRES5.*											
33 ·	N/A [1]	N/A [1]	N/A [1]	ALTER	ALTER	YES	**NO**	FAIL	FAIL	N	N
34 ·	N/A [1]	N/A [1]	N/A [1]	NONE	ALTER	YES	**NO**	FAIL	FAIL	N	N
35 ✍	N/A [1]	N/A [1]	N/A [1]	ALTER	NONE	YES	**NO**	FAIL	FAIL	N	N
36 ✍	N/A [1]	N/A [1]	N/A [1]	NONE	NONE	YES	**NO**	FAIL	FAIL	N	N

Test case/ success	RACF settings					TAPEAUTH settings in the DEVSUPnn				DFSMSrmm [2]	
	TAPEVOL	TAPEDSN	THM00n[4]	MHLRES1.**	MHLRES7.**	DSN	F1	RC4	RC8	TPRACF	RACF

[1] N/A means that this function is not activated at this time or no TAPEVOL profile exists.
[2] The DFSMSrmm settings are stored in the current active DFSMSrmm EDGRMMnn PARMLIB member.
In this table, we show only the tape security related to DFSMSrmm operands we have modified for our test cases.
TPRACF is an operand of the OPTION command and RACF is an option of the VLPOOL command.
[3] "A" is the abbreviation of the TPRACF(AUTOMATIC) processing.
[4] THM001 is the TAPEVOL profile protecting the volume we used to create the two data sets for test cases 1 to 19.

Test case 1

The first test case has the following characteristics:

Function Create two new tape data sets on a volume in status scratch. The IEBGENER program is used to copy a member of a library.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors. DFSMSrmm created a new RACF TAPEVOL profile for volume THM001 and the user MHLRES5 has ALTER access to it.

Example 13-17 shows the JCL and the settings that we used to create the two data sets.

Example 13-17 Sample JCL used for test case 1

```
//*    ****************************************************************
//*    * /F DFRMM,M=R2                                                *
//*    ****************************************************************
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES1.RACF.TEST1.FILE1 NONVSAM NOSCRATCH
```

```
 DELETE MHLRES7.RACF.TEST1.FILE2 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD   DISP=(,CATLG),DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,99),
//         UNIT=ATL3
//SYSIN    DD   DUMMY
```

Example 13-18 shows the most important results of the RACF commands.

Note: All the RACF command result examples have been modified so that the AUDITING, NOTIFY, INSTALLATION DATA, and SECURITY LEVEL information is not shown.

Example 13-18 RACF command results of test case 1

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES5.**') ALL
INFORMATION FOR DATASET MHLRES5.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES5         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 2

The second test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors, because the user has ALTER access to the TAPEVOL profile and to the two profiles in class DATASET.

Example 13-19 shows the JCL and the settings that we used to read the two data sets.

Example 13-19 Sample JCL used for test 2

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACC(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-20 shows the most important results of the RACF commands.

Example 13-20 RACF commands for test case 2

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
MHLRES5   ALTER       000000
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 3

The third test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, the user has no access to the first data set. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors because the user has ALTER access to the TAPEVOL profile and to the second data set. This scenario is only possible because after RACF checks the access to the data set profile, RACF checks the access to the TAPEVOL profile. If either one of the two checks ended with a return code zero, the access is allowed.

Example 13-21 on page 507 shows the JCL and the settings that we used to read the two data sets.

Example 13-21 Sample JCL used for test 3

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACC(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-22 shows the most important results of the RACF commands.

Example 13-22 RACF commands for test case 3

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
```

```
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
MHLRES5   ALTER       000000
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 4

The fourth test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, the user has no access to the TAPEVOL profile. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors, because the user has ALTER access to the data set profiles and there is no check for the access to the TAPEVOL profile.

Example 13-23 shows the JCL and the settings that we used to read the two data sets.

Example 13-23 Sample JCL used for test 4

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(NONE)
```

```
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(NONE)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-24 shows the most important results of the RACF commands.

Example 13-24 RACF commands for test case 4

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE               NONE    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
MHLRES5   NONE        000000
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO
```

```
YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 5

The fifth test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, the user has no access to the TAPEVOL profile and no access to the first file. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result In this case, we get a security violation for the first data set, because we have no access to the data set profile or to the TAPEVOL profile. The second data set can be read without any errors.

Example 13-25 shows the JCL and the settings that we used to read the two data sets.

Example 13-25 Sample JCL used for test 5

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(NONE)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(NONE)
 SETR REFRESH RACLIST(TAPEVOL)
```

```
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-26 shows the most important results of the RACF commands.

Example 13-26 RACF commands for test case 5

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE               NONE    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
MHLRES5   NONE        000000
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
```

```
    ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-27, you can see that we get a security violation for file one.

Example 13-27 Test case 5 job output

```
....
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  THM001 CL(TAPEVOL )
  INSUFFICIENT ACCESS AUTHORITY
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST1.FILE1 CL(DATASET ) VOL(THM001)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-60,IFG0194F,RACFTS05,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST1.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000060
 TIME=18.11.17  SEQ=00982  CPU=0000  ASID=0039
IEF472I RACFTS05 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000060
....
```

Test case 6

The sixth test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, the RACF CLASS TAPEVOL is inactive. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors, because the user has ALTER access to both data set profiles. The RACF TAPEVOL profile is not checked.

Example 13-28 shows the JCL and the settings that we used to read the two data sets.

Example 13-28 Sample JCL used for test 6

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            NO ACCESS
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
```

```
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) DELETE
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-29 shows the most important results of the RACF commands.

Example 13-29 RACF commands for test case 6

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR TAPEDSN
WARNING: TAPEDSN OPTION ACTIVE, TAPEVOL CLASS IS NOT ACTIVE
READY

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO
```

```
YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 7

The seventh test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, the RACF CLASS TAPEVOL is inactive and the user has no access to the first data set. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result In this case, we get a security violation for the first data set, because we have no access to the data set profile and the TAPEVOL profile is not checked because RACF CLASS TAPEVOL is inactive. The second data set can be read without any errors.

Example 13-30 shows the JCL and the settings that we used to read the two data sets.

Example 13-30 Sample JCL used for test 7

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            NO ACCESS
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
```

```
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-31 shows the most important results of the RACF commands.

Example 13-31 RACF commands for test case 7

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR TAPEDSN
WARNING: TAPEDSN OPTION ACTIVE, TAPEVOL CLASS IS NOT ACTIVE
READY

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
```

```
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-32, you can see the security violation that we get for the first file, but in this case only for the data set profile.

Example 13-32 Test case 7 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST1.FILE1 CL(DATASET ) VOL(THM001)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-60,IFG0194F,RACFTS07,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST1.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000060
 TIME=18.13.50  SEQ=00983  CPU=0000  ASID=0039
IEF472I RACFTS07 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000060
....
```

Test case 8

The eighth test case has the following characteristics:

Function Read the two previously created tape sets. But in this case, both RACF CLASSES TAPEVOL and TAPEDSN are inactive. The IEBGENER program is used to read the files. In this case, you have all necessary RACF access and the job ended without any error, as expected.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors, because there are no security checks.

Example 13-33 shows the JCL and the settings that we used to read the two data sets.

Example 13-33 Sample JCL used for test 8

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           INACTIVE
//*        THM001            NO ACCESS
```

```
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-34 shows the most important results of the RACF commands.

Example 13-34 RACF commands for test case 8

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR NOTAPEDSN

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO
```

```
YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job that is shown in Example 13-33 on page 516, you can see that there was no security violation.

Test case 9

The ninth test case has the following characteristics:

Function Read the two previously created tape sets. Both RACF CLASSES TAPEVOL and TAPEDSN are inactive and the user has no access to the data set profile that protects the second file. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE1

Result The job ended without any errors, because there are no security checks.

Example 13-35 shows the JCL and the settings that we used to read the two data sets.

Example 13-35 Sample JCL used for test 9

```
//*    RMM OPTIONS:
//*        TPRACF(A)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            NO ACCESS
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           INACTIVE
//*    FUNCTION
//*        READ THE PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
```

```
 SETR NOTAPEDSN
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-36 shows the most important results of the RACF commands.

Example 13-36 RACF commands for test case 9

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR NOTAPEDSN

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000001

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
```

```
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

Test case 10

The tenth test case has the following characteristics:

Function Create an additional third tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.

Data set name MHLRES7.RACF.TEST1.FILE3

Result The job ended without any errors because we have ALTER access to the RACF TAPEVOL profile for volume THM001 and the user MHLRES5 has ALTER access to the data set profile.

Example 13-37 shows the JCL and the settings that we used to create the third data set.

Example 13-37 Sample JCL used for test 10

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST1.FILE3
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE3 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
```

```
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(03,SL),DSN=MHLRES7.RACF.TEST1.FILE3,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
//SYSIN    DD   DUMMY
```

Example 13-38 shows the most important results of the RACF commands.

Example 13-38 RACF commands for test case 10

```
 RL TAPEVOL THM001
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 11

The eleventh test case has the following characteristics:

Function Create an additional fourth tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.

Data set names MHLRES7.RACF.TEST1.FILE4

Result In this case, we get a security violation for the fourth data set that we tried to create, because we have no access to the fourth data set profile.

Example 13-39 shows the JCL and the settings that we used to create the fourth data set.

Example 13-39 Sample JCL used for test 11

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(NONE)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SETS
//*            MHLRES7.RACF.TEST1.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST1.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
//SYSIN    DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-40 on page 523 shows the most important results of the RACF commands.

Example 13-40 RACF commands for test case 11

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-41, you can see that we get a security violation for file four.

Example 13-41 Test case 11 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST1.FILE4 CL(DATASET ) VOL(THM001)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-60,IFG0194F,RACFTS11,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST1.FILE4
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000060
 TIME=08.00.54  SEQ=01017  CPU=0000  ASID=002C
IEF472I RACFTS11 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000060
....
```

Test case 12

The twelfth test case has the following characteristics:

Function Repeat the creation of the additional fourth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.

Data set name MHLRES7.RACF.TEST1.FILE4

Result The job ended without any errors, because we have access to the RACF TAPEVOL profile for volume THM00, although the user MHLRES5 does not have ALTER access to the data set profile. This access is not checked because the RACF class TAPEDSN is inactive.

Example 13-42 shows the JCL and the settings that we used to create the fourth data set.

Example 13-42 Sample JCL used for test 12

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           INACTIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST1.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE THM001CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST1.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
```

```
//SYSIN    DD   DUMMY
```

Example 13-43 shows the most important results of the RACF commands.

Example 13-43 RACF commands for test case 12

```
 RL TAPEVOL THM001ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 13

The thirteenth test case has the following characteristics:

Function	Create a fifth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.
Data set name	MHLRES7.RACF.TEST1.FILE4
Result	The job ended without any errors because the RACF option NOTAPEDSN class is being set and the user MHLRES5 has ALTER access to the TAPEVOL.

Example 13-44 shows the JCL and the settings we used to create the fifth data set.

Example 13-44 Sample JCL used for test 13

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
```

```
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           INACTIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SETS
//*            MHLRES7.RACF.TEST1.FILE5
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE THM001CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE5 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(05,SL),DSN=MHLRES7.RACF.TEST1.FILE5,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
//SYSIN    DD   DUMMY
```

Example 13-45 shows the most important results of the RACF commands.

Example 13-45 RACF commands for test case 13

```
 RL TAPEVOL THM001ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
```

```
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

Test case 14

The fourteenth test case has the following characteristics:

Function Create a sixth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.

Data set name MHLRES7.RACF.TEST1.FILE6

Result The job ended without any errors because the user MHLRES5 has ALTER access to the data set profile and RACF class TAPEVOL is inactive. Example 13-46 shows the JCL that we used.

Example 13-46 Sample JCL used for test 14

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*        THM001           ACC(ALTER)
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SETS
//*            MHLRES7.RACF.TEST1.FILE6
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
```

```
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE6 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(06,SL),DSN=MHLRES7.RACF.TEST1.FILE6,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
//SYSIN    DD   DUMMY
```

Example 13-47 shows the most important results of the RACF commands.

Example 13-47 RACF commands

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR TAPEDSN
WARNING: TAPEDSN OPTION ACTIVE, TAPEVOL CLASS IS NOT ACTIVE

 RL TAPEVOL THM001ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

```
 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 15

The fifteenth test case has the following characteristics:

Function Create a seventh new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library.

Data set name MHLRES7.RACF.TEST1.FILE7

Result In this case, we get a security violation for the seventh data set that we want to create because we have no access to the data set profile and the RACF CLASS TAPEVOL is inactive.

Example 13-48 shows the JCL and the settings that we used to create the seventh data set.

Example 13-48 Sample JCL used for test 15

```
//*    RMM OPTIONS:
//*        TPRACF(Y)
//*        RACF(Y)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(NONE)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SETS
//*            MHLRES7.RACF.TEST1.FILE7
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
```

```
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST1.FILE6 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(07,SL),DSN=MHLRES7.RACF.TEST1.FILE7,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM001
//SYSIN    DD   DUMMY
```

Example 13-49 shows the most important results of the RACF commands.

Example 13-49 RACF commands for test case 15

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR TAPEDSN
WARNING: TAPEDSN OPTION ACTIVE, TAPEVOL CLASS IS NOT ACTIVE
READY
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
READY
 SETR REFRESH RACLIST(TAPEVOL)
RACLIST REFRESH of class TAPEVOL ignored. The class is not active yet.
SETROPTS command complete.

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
```

```
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-50, you can see that we get a security violation for file seven.

Example 13-50 Test case 15 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST1.FILE7 CL(DATASET ) VOL(THM001)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-60,IFG0194F,RACFTS15,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST1.FILE7
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000060
 TIME=08.49.02  SEQ=01037  CPU=0000  ASID=0039
IEF472I RACFTS15 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000060
....
```

Test case 16

The sixteenth test case has the following characteristics:

Function Read the six previously created tape sets, but in this case, both RACF CLASSES TAPEVOL and TAPEDSN are inactive. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE2
MHLRES7.RACF.TEST1.FILE3
MHLRES7.RACF.TEST1.FILE4
MHLRES7.RACF.TEST1.FILE5
MHLRES7.RACF.TEST1.FILE6

Result The job ended without any errors, because no security checks are being performed.

Example 13-35 on page 518 shows the JCL and the settings that we used to read the two data sets.

Example 13-51 Sample JCL used for test 16

```
//*    ****************************************************************
//*    * /F DFRMM,M=02                                                *
//*    ****************************************************************
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
```

```
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ ALL PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*            MHLRES7.RACF.TEST1.FILE3
//*            MHLRES7.RACF.TEST1.FILE4
//*            MHLRES7.RACF.TEST1.FILE5
//*            MHLRES7.RACF.TEST1.FILE6
//*            MHLRES7.RACF.TEST1.FILE7
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE3,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(3,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP04   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE4,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(4,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP05   EXEC PGM=IEBGENER
```

```
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE5,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(5,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP06   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE6,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(6,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP07   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE7,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(7,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-52 shows the most important results of the RACF commands.

Example 13-52 RACF commands for test case 16

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 17

The seventeenth test case has the following characteristics:

Function: Read the six previously created tape sets. But in this case, RACF CLASS TAPEVOL is active and the option TAPEDSN is being set. The IEBGENER program is used to read the files.

Data set names:
MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE2
MHLRES7.RACF.TEST1.FILE3
MHLRES7.RACF.TEST1.FILE4
MHLRES7.RACF.TEST1.FILE5
MHLRES7.RACF.TEST1.FILE6

Result: The job ended without any errors, although the user has no access to the data set profile MHLRES1.**. In this situation, RACF checks that the user has UPDATE access to the TAPEVOL profile.

Example 13-53 shows the JCL and the settings that we used to read the two data sets.

Example 13-53 Sample JCL used for test 17

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           ACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ ALL PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*            MHLRES7.RACF.TEST1.FILE3
//*            MHLRES7.RACF.TEST1.FILE4
//*            MHLRES7.RACF.TEST1.FILE5
//*            MHLRES7.RACF.TEST1.FILE6
//*            MHLRES7.RACF.TEST1.FILE7
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR CLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
```

```
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE3,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(3,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP04   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE4,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(4,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP05   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE5,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(5,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP06   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE6,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(6,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP07   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE7,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(7,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-54 shows the most important results of the RACF commands.

Example 13-54 RACF commands for test case 17

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001
```

```
LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 18

The eighteenth test case has the following characteristics:

Function Read the seven previously created tape sets. However, in this case, the RACF CLASS TAPEVOL is inactive and option TAPEDSN is being set. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE2
MHLRES7.RACF.TEST1.FILE3
MHLRES7.RACF.TEST1.FILE4
MHLRES7.RACF.TEST1.FILE5
MHLRES7.RACF.TEST1.FILE6

Result The job ended without any errors because the user has access to the data set profile and the RACF TAPEVOL class is inactive.

Example 13-55 shows the JCL and the settings that we used to read the previously created data sets.

Example 13-55 Sample JCL used for test 18

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
```

```
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ ALL PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*            MHLRES7.RACF.TEST1.FILE3
//*            MHLRES7.RACF.TEST1.FILE4
//*            MHLRES7.RACF.TEST1.FILE5
//*            MHLRES7.RACF.TEST1.FILE6
//*            MHLRES7.RACF.TEST1.FILE7
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE3,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(3,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP04   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE4,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(4,SL)
//SYSUT2   DD   DUMMY
```

```
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP05   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE5,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(5,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP06   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE6,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(6,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP07   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE7,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(7,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-56 shows the most important results of the RACF commands.

Example 13-56 RACF commands for test case 18

```
 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
```

```
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

13.6.1 Test case 19

The nineteenth test case has the following characteristics:

Function Read the six previously created tape sets. But in this case, the RACF CLASS TAPEVOL is inactive and option TAPEDSN is being set. The IEBGENER program is used to read the files.

Data set names MHLRES1.RACF.TEST1.FILE1
MHLRES7.RACF.TEST1.FILE2
MHLRES7.RACF.TEST1.FILE3
MHLRES7.RACF.TEST1.FILE4
MHLRES7.RACF.TEST1.FILE5
MHLRES7.RACF.TEST1.FILE6

Result We get a security violation, because we have no access to the data set profile MHLRES1.** and the RACFCLASS TAPEVOL is inactive.

Example 13-57 shows the JCL and the settings that we used to read the previously created data sets.

Example 13-57 Sample JCL used for test 19

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=N/A
//*        TAPEAUTHF1=N/A
//*        TAPEAUTHRC4=N/A
//*        TAPEAUTHRC8=N/A
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        THM001            ACC(ALTER)
//*        TAPEVOL           INACTIVE
//*        TAPEDSN           ACTIVE
//*    FUNCTION
//*        READ ALL PREVIOUSLY CREATED DATA SETS
//*            MHLRES1.RACF.TEST1.FILE1
//*            MHLRES7.RACF.TEST1.FILE2
//*            MHLRES7.RACF.TEST1.FILE3
//*            MHLRES7.RACF.TEST1.FILE4
//*            MHLRES7.RACF.TEST1.FILE5
//*            MHLRES7.RACF.TEST1.FILE6
//*            MHLRES7.RACF.TEST1.FILE7
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR TAPEDSN
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
 SETR REFRESH RACLIST(TAPEVOL)
 RL TAPEVOL THM001 ALL
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
```

```
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR REFRESH GENERIC(DATASET)
 SETROPTS LIST
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST1.FILE1,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE2,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP03   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE3,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(3,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP04   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE4,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(4,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP05   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE5,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(5,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP06   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE6,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(6,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP07   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST1.FILE7,
//         VOL=(,RETAIN,,,SER=(THM001)),LABEL=(7,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-58 on page 541 shows the most important results of the RACF commands.

Example 13-58 RACF commands

```
 SETR NOCLASSACT(TAPEVOL)
READY
 SETR TAPEDSN
WARNING: TAPEDSN OPTION ACTIVE, TAPEVOL CLASS IS NOT ACTIVE
READY
 PE THM001 CLASS(TAPEVOL) ID(MHLRES5) ACCESS(ALTER)
READY
 SETR REFRESH RACLIST(TAPEVOL)
RACLIST REFRESH of class TAPEVOL ignored. The class is not active yet.
SETROPTS command complete.

 RL TAPEVOL THM001 ALL
CLASS      NAME
-----      ----
TAPEVOL    THM001

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    MHLRES5         NONE              ALTER    NO

USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
STC       ALTER       000005
MHLRES5   ALTER       000000

 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-59, you can see that we get a security violation for file one.

Example 13-59 Test case 19 job output

```
ICH70001I MHLRES5  LAST ACCESS AT 18:37:04 ON SUNDAY, AUGUST 5, 2007
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST1.FILE1 CL(DATASET ) VOL(THM001)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
```

```
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-60,IFG0194F,RACFTS19,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST1.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000060
 TIME=18.39.36  SEQ=00996  CPU=0000  ASID=0039
IEF472I RACFTS19 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000060
....
```

Test case 20

This test case and all following test cases are under the conditions that we have specified in the TAPEAUTH settings in the DEVSUPnn member. Both RACF CLASSES TAPEVOL and TAPEDSN are inactive.

The twentieth test case has the following characteristics:

Function Create two new tape sets on a tape volume in scratch status. The IEBGENER program is used to write the files. TAPEAUTHDSN and TAPEAUTHF1 are set and the option for TAPEAUTHRC4 and TAPEAUTHRC8 is set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result The job ended without any errors, because the user has access to the data set profiles.

Example 13-60 shows the JCL and the settings that we used to create the two data sets.

Example 13-60 Sample JCL used for test 20

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES5.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES5.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES5.**') ALL
```

```
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES1.RACF.TEST2.FILE1 NONVSAM NOSCRATCH
 DELETE MHLRES7.RACF.TEST2.FILE2 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD   DISP=(,CATLG),DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,99),
//         UNIT=ATL3
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP1    EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(02,SL),DSN=MHLRES7.RACF.TEST2.FILE2,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=(,RETAIN,,REF=*.STEP01.SYSUT2)
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
```

Test case 21

The twenty-first test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN and TAPEAUTHF1 are set and the option for TAPEAUTHRC4 and TAPEAUTHRC8 is set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result The job ended without any errors, because the user has access to the data set profiles.

Example 13-61 shows the JCL that we used for test case 21.

Example 13-61 Sample JCL used for test 21

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
```

```
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-62 shows the most important results of the RACF commands.

Example 13-62 Test case 21 job output

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 22

The twenty-second test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN and TAPEAUTHF1 are set and the option for TAPEAUTHRC4 and TAPEAUTHRC8 is set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result We received two security violations because the user has no access to the data set profile MHLRES1.** that protects the first file on the volume. So, we were unable to access any data set on the volume.

Example 13-63 shows the JCL that we used for test case 22.

Example 13-63 Sample JCL used for test 22

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-64 shows the list of the RACF profile in the class TAPEVOL.

Example 13-64 List of the RACF profile in class TAPEVOL

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)
```

```
LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-65, you can see the security violations that we received for the first and second files.

Example 13-65 Test case 22 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS22,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST2.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.34.36  SEQ=00892  CPU=0000  ASID=0039
IEF472I RACFTS22 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS22,STEP02,SYSUT1,0B23,,MHLRES7.RACF.TEST2.FILE2
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.35.22  SEQ=00893  CPU=0000  ASID=0039
IEF472I RACFTS22 STEP02 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 23

The twenty-third test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. The TAPEAUTHDSN and TAPEAUTHF1 options are specified in the DEVSUPxx member of SYS1.PARMLIB and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result Although we have access to the first file on the volume, we received a security violation because the user has no access to the data set profile MHLRES7.** to read the second file.

Example 13-66 shows the JCL that we used for test case 23.

Example 13-66 Sample JCL used for test 23

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-67 shows the most important results of the RACF commands.

Example 13-67 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)
```

```
LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-68, you can see the security violation that we got for the second file.

Example 13-68 Test case 23 job output

```
...
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE2 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS23,STEP02,SYSUT1,0B23,,MHLRES7.RACF.TEST2.FILE2
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.36.09  SEQ=00894  CPU=0000  ASID=002C
IEF472I RACFTS23 STEP02 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 24

The twenty-fourth test case has the following characteristics:

Function	Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN and TAPEAUTHF1 are set and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.
Data set names	MHLRES1.RACF.TEST2.FILE1 MHLRES7.RACF.TEST2.FILE2
Result	We received two security violations because the user has no access to the data set profiles MHLRES1.** and MHLRES7.**.

Example 13-69 shows the JCL we used for test case 24.

Example 13-69 Sample JCL used for test 24

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
```

```
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-70 shows the most important results of the RACF commands.

Example 13-70 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
```

```
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-71, you can see that we received two security violations: one security violation for file 1 and one security violation for file 2.

Example 13-71 Test case 24 job output

```
...
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS24,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST2.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=10.40.55  SEQ=01052  CPU=0000  ASID=002C
IEF472I RACFTS24 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE2 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS24,STEP02,SYSUT1,0B23,,MHLRES7.RACF.TEST2.FILE2
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=10.41.42  SEQ=01053  CPU=0000  ASID=002C
....
```

Test case 25

The twenty-fifth test case has the following characteristics:

Function Create an additional third new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN and TAPEAUTHF1 are set and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE3

Result The job ended without any errors, because we have access to both data set profiles, protecting the first file on the volume and the new file that we are creating.

Example 13-72 shows the JCL and the settings that we used to create the third data set.

Example 13-72 Sample JCL used for test 25

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
```

```
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE3
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES5.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE3 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(03,SL),DSN=MHLRES7.RACF.TEST2.FILE3,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-73 shows the most important results of the RACF commands.

Example 13-73 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
```

```
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 26

The twenty-sixth test case has the following characteristics:

Function Create a fourth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN and TAPEAUTHF1 are set and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE4

Result We received a security violation. The file is not created because we have no access to the RACF data set profile MHLRES1.** that protects the first file on the volume.

Example 13-74 shows the JCL that we used for test case 26.

Example 13-74 Sample JCL used for test 26

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
```

```
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST2.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-75 shows the most important results of the RACF commands.

Example 13-75 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-76, you can see that we received a security violation although the data set profile protects the first file on the volume.

Example 13-76 Test case 26 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS26,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST2.FILE4
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.39.27  SEQ=00897  CPU=0000  ASID=0039
IEF472I RACFTS26 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 27

The twenty-seventh test case has the following characteristics:

Function Re-create a fourth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN and TAPEAUTHF1 are set and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE4

Result We received a security violation and the file is not created because we have no access to the RACF data set profile MHLRES7.** that protects the data set that we want to create.

Example 13-77 shows the JCL that we used for test case 27.

Example 13-77 Sample JCL used for test 27

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST2.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//SUBMIT   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(RACF28)
//SYSUT2   DD   SYSOUT=(A,INTRDR)
//SYSIN    DD   DUMMY
```

Example 13-78 on page 555 shows the most important results of the RACF commands.

Example 13-78 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-79, you can see that there was no security violation reading the first file. However, we were unable to read the second file because we have no access to the profile that protects this file.

Example 13-79 Test case 27 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE4 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS27,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST2.FILE4
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.40.27  SEQ=00898  CPU=0000  ASID=002C
 IEF472I RACFTS27 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 28

The twenty-eighth test case has the following characteristics:

Function Re-create a fourth new tape data set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN and TAPEAUTHF1 are set and the options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE4

Result We received a security violation because we have no access to the RACF data set profile MHLRES1.** that protects the first file on the volume nor to the profile MHLRES7.** that protects the data set that we want to create.

Example 13-80 on page 556 shows the JCL that we used for test case 28.

Example 13-80 Sample JCL used for test 28

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST2.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-81 shows the most important results of the RACF commands.

Example 13-81 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

```
 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-82, you can see that we were unable to create the new data set because of the security violation for the data set profile.

Example 13-82 Test case 28 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE4 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS28,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST2.FILE4
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.41.27  SEQ=00899  CPU=0000  ASID=002C
IEF472I RACFTS28 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 29

The twenty-ninth test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result The job ended without any errors because the user has access to both data set profiles protecting these files.

Example 13-83 shows the JCL that we used for test case 29.

Example 13-83 Sample JCL used for test 29

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.***       ACC(ALTER)
//*        TAPEVOL           INCATIVE
```

```
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        Read the two DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-84 shows the most important results of the RACF commands.

Example 13-84 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 30

The thirtieth test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result We received a security violation because the user has no access to the data set profile MHLRES1.**. The second file can be read without any errors.

Example 13-85 shows the JCL that we used for test case 30.

Example 13-85 Sample JCL used for test 30

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
```

```
//SYSIN    DD   DUMMY
```

Example 13-86 shows the most important results of the RACF commands.

Example 13-86 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

In the output of the job in Example 13-87, you can see that there was a security violation for the first file.

Example 13-87 Test case 30 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS30,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST2.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.47.20  SEQ=00901  CPU=0000  ASID=0039
IEF472I RACFTS30 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 31

The thirty-first test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result We received a security violation because the user has no access to the data set profile MHLRES7.**. The first file can be read without any errors.

Example 13-88 shows the JCL that we used for test case 31.

Example 13-88 Sample JCL used for test 31

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
```

Example 13-89 shows the most important results of the RACF commands.

Example 13-89 Result of the RACF commands

```
  LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
```

```
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-90, you can see that we received a security violation for the data set MHLRES7.RACF.TEST2.FILE2.

Example 13-90 Test case 31 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE2 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS31,STEP02,SYSUT1,0B23,,MHLRES7.RACF.TEST2.FILE2
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=16.49.21  SEQ=00902  CPU=0000  ASID=002C
IEF472I RACFTS31 STEP02 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 32

The thirty-second test case has the following characteristics:

Function Read the two previously created tape sets. The IEBGENER program is used to read the files. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set names MHLRES1.RACF.TEST2.FILE1
MHLRES7.RACF.TEST2.FILE2

Result We received a security violation because the user has no access to the data set profiles MHLRES1.** and MHLRES7.**. No file can be read.

Example 13-91 shows the JCL that we used for test case 32.

Example 13-91 Sample JCL used for test 32

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
```

```
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE TWO NEW DATA SETS
//*            MHLRES1.RACF.TEST2.FILE1
//*            MHLRES7.RACF.TEST2.FILE2
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES1.RACF.TEST2.FILE1,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(1,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
//STEP02   EXEC PGM=IEBGENER,COND=EVEN
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES7.RACF.TEST2.FILE2,
//         VOL=(,RETAIN,,,SER=(THM002)),LABEL=(2,SL)
//SYSUT2   DD   DUMMY
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
```

Example 13-92 shows the most important results of the RACF commands.

Example 13-92 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO
```

```
YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-93, you can see that we received a security violation for both data set profiles protecting these files.

Example 13-93 Test case 32 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES1.RACF.TEST2.FILE1 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES1.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS32,STEP01,SYSUT1,0B23,,MHLRES1.RACF.TEST2.FILE1
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=11.52.12  SEQ=01054  CPU=0000  ASID=002C
IEF472I RACFTS32 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE2 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS32,STEP02,SYSUT1,0B23,,MHLRES7.RACF.TEST2.FILE2
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=11.53.03  SEQ=01055  CPU=0000  ASID=002C
IEF472I RACFTS32 STEP02 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
....
```

Test case 33

The thirty-third test case has the following characteristics:

Function Create a fourth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE5

Result The job ended without any errors because we have access to both data set profiles protecting the first file on the volume and the new file that we are creating.

Example 13-94 shows the JCL that we used for test case 33.

Example 13-94 Sample JCL used for test 33

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
```

```
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE4
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES5.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE4 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(04,SL),DSN=MHLRES7.RACF.TEST2.FILE4,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-95 shows the most important results of the RACF commands.

Example 13-95 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO
```

```
YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 34

The thirty-fourth test case has the following characteristics:

Function Create a fifth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE5

Result The job ended without any errors, although we have no access to the first file on the volume. This is not checked, because the option TAPEAUTHF1 is not set to yes.

Example 13-96 shows the JCL that we used for test case 34.

Example 13-96 Sample JCL used for test 34

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(ALTER)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE5
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(ALTER) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE5 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
```

```
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(05,SL),DSN=MHLRES7.RACF.TEST2.FILE5,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-97 shows the most important results of the RACF commands.

Example 13-97 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM
```

Test case 35

The thirty-fifth test case has the following characteristics:

Function Create a sixth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE5

Result We received a security violation and the file is not created because we have no access to the RACF data set profile MHLRES7.** that protects the data set that we want to create.

Example 13-98 shows the JCL that we used for test case 35.

Example 13-98 Sample JCL used for test 35

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(ALTER)
```

```
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE6
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(ALTER) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE6 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(06,SL),DSN=MHLRES7.RACF.TEST2.FILE6,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
```

Example 13-99 shows the most important results of the RACF commands.

Example 13-99 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
   ALTER        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-100, you can see that we received a security violation for the data set profile MHLRES7.**.

Example 13-100 Test case 35 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE6 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS35,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST2.FILE6
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=17.20.50  SEQ=00905  CPU=0000  ASID=002C
IEF472I RACFTS35 STEP01 - COMPLETION CODE - SYSTEM=913 USER=0000 REASON=00000038
```

Test case 36

The thirty-sixth test case has the following characteristics:

Function Re-create the sixth new tape set on the previously used tape volume. The IEBGENER program is used to copy a member of a library. TAPEAUTHDSN is set to YES but TAPEAUTHF1 is set to NO. The options for TAPEAUTHRC4 and TAPEAUTHRC8 are set to FAIL.

Data set name MHLRES7.RACF.TEST2.FILE5

Result We received a security violation and the file is not created, because we have no access to the RACF data set profile MHLRES7.** that protects the data set that we want to create. The first file protection is not checked because the TAPEAUTHF1 checking is not set.

Example 13-101 shows the JCL we used for test case 36.

Example 13-101 Sample JCL used for test 36

```
//*    RMM OPTIONS:
//*        TPRACF(N)
//*        RACF(N)
//*    DEVSUP
//*        TAPEAUTHDSN=YES
//*        TAPEAUTHF1=YES
//*        TAPEAUTHRC4=FAIL
//*        TAPEAUTHRC8=FAIL
//*    RACF
//*        MHLRES1.**        ACC(NONE)
//*        MHLRES7.**        ACC(NONE)
//*        TAPEVOL           INCATIVE
//*        TAPEDSN           INCATIVE
//*    FUNCTION
//*        CREATE AN ADDITIONAL NEW DATA SET
//*            MHLRES7.RACF.TEST2.FILE6
//*
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 SETR NOCLASSACT(TAPEVOL)
 SETR NOTAPEDSN
 PE 'MHLRES1.**' ACC(NONE) ID(MHLRES5)
 PE 'MHLRES7.**' ACC(NONE) ID(MHLRES5)
```

```
 SETR REFRESH GENERIC(DATASET)
 LD DATASET('MHLRES1.**') ALL
 LD DATASET('MHLRES7.**') ALL
 SETR LIST
/*
//CLEANUP  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DELETE MHLRES7.RACF.TEST2.FILE6 NONVSAM NOSCRATCH
 SET MAXCC=0
/*
//*-------------------------------------------------------------*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   DISP=SHR,DSN=MHLRES5.RACF.CNTL(TEXT)
//SYSUT2   DD LABEL=(06,SL),DSN=MHLRES7.RACF.TEST2.FILE6,
//         DISP=(,CATLG,DELETE),UNIT=ATL3,RETPD=02,
//         VOLUME=SER=THM002
//SYSIN    DD   DUMMY
//*-------------------------------------------------------------*
```

Example 13-102 shows the most important results of the RACF commands.

Example 13-102 Result of the RACF commands

```
 LD DATASET('MHLRES1.**') ALL
INFORMATION FOR DATASET MHLRES1.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES1         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM

 LD DATASET('MHLRES7.**') ALL
INFORMATION FOR DATASET MHLRES7.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ----------------   -------   -----
 00    MHLRES7         NONE          NO      NO

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    NONE        SYS1           NON-VSAM
```

In the output of the job in Example 13-103, you can see that we received a security violation for the data set profile MHLRES7.** only.

Example 13-103 Test case 36 job output

```
....
IEF237I DMY  ALLOCATED TO SYSIN
ICH408I USER(MHLRES5 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)
  MHLRES7.RACF.TEST2.FILE6 CL(DATASET ) VOL(THM002)
  INSUFFICIENT ACCESS AUTHORITY
  FROM MHLRES7.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194F,RACFTS36,STEP01,SYSUT2,0B23,,MHLRES7.RACF.TEST2.FILE6
```

```
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=913  REASON CODE=00000038
 TIME=17.21.52  SEQ=00906  CPU=0000  ASID=002C
 ....
```

14

Report Generator

This chapter describes how you can use the Data Facility System Managed Storage removable media manager (DFSMSrmm) Report Generator to create customized reports. The DFSMSrmm Report Generator is an Interactive System Productivity Facility (ISPF) application that you use to create reports to show the status of the resources that DFSMSrmm manages for you. These resources include volumes, data sets, racks, owners, and the retention and movement policies that are established for your installation.

This chapter includes the following information:

- Report Generator overview
- Setting up the Report Generator for your installation
- Create a report

At the end of this chapter, you will be familiar with the use of the DFSMSrmm Report Generator.

14.1 Report Generator overview

The DFSMSrmm Report Generator is an ISPF application that you can use to create reports. It provides the following functions:

- Provides reports that you can run as-is or that you can modify as you want. You can use samples to create reports for volumes, data sets, racks, owners, and the retention and movement policies that are established for your installation. You can modify these samples to create tailored reports. DFSMSrmm ships samples in SYS1.SAMPLIB.
- Generates job control language (JCL) that is based on specifications that you use to submit the report jobs. The generation of JCL depends on the report type and therefore the macros that map the data records. The generation knows, based on the macro name and keyword options used, whether to generate a DCOLLECT job step, a DFSMShsm FSR reformat, a DFSMSrmm extract, or a copy of System Management Facilities (SMF) records.
- Includes samples for reporting from DCOLLECT and DFSMShsm data.

The DFSMSrmm Report Generator is updated to support keywords for assembler macros from which report types are derived, and to add new built-in data extract steps. This is in support of new SMF record types from DFSMSrmm, and for DFSMShsm and DCOLLECT reporting.

Any new report type or report definitions that are created on z/OS V1R10, which include macro keyword information, can be used by a lower-level release. However, the macro keywords are ignored and are removed if the report type or report definition is changed or updated. The original input report type or report definition is not affected and can be reused later on a supporting release to use the macro keywords.

Suggestion: Use z/OS V1R10 or a later release to update and customize report types and reports that require you to specify macro keywords. When the report JCL is generated, you can run that JCL on any supported release.

14.2 Setting up the Report Generator for your installation

The following steps must be used to set up the DFSMSrmm Report Generator before you can use it. If you have defined the DFSMSrmm Report Generation settings, you can skip this section and go directly to 14.3, “Create a report” on page 581.

1. Customize the EDGRMAIN EXEC. The default name of the library where you can find this REXX Exec is SYS1.SEDGEXE1. The REXX variable names that you can customize all start with the characters `cedggrdl`.

   a. Define the installation library name and optionally customize the product library name in EXEC EDGRMAIN. There is no installation library name in the EXEC, so you must add the name.

   b. Update the default naming convention in EXEC EDGRMAIN for the user library name and the JCL library name, if necessary.

   Figure 14-1 on page 575 shows the section of the EDGRMAIN EXEC that you can customize. For example, to allow a shared library that is used by all users, you must remove the four lines in strike-through text in the sample and add the highlighted line.

In our example, we use two libraries:

– RMM.REPORT.LIB
– RMM.REPORT.JCL

```
/* Initialise Report library names */                          /*§09A*/
address "ISPEXEC" "VGET ZPREFIX"                               /*§09A*/
If length(zprefix) = 0 then                                    /*§10C*/
  edggpref = sysvar('SYSUID')                                  /*§09A*/
else                                                           /*§10C*/
  edggpref = zprefix                                           /*§10C*/

  edggpref = “RMM”                                             /*NSCH*/

cedggrdlu = "'"edggpref!!".REPORT.LIB'"  /* User Library          §10C*/
cedggrdlj = "'"edggpref!!".REPORT.JCL'"  /* User JCL Library      §10C*/
cedggrdlp = "'SYS1.SAMPLIB'"             /* Product Library       §10C*/
cedggrdli = ""                           /* Installation Library  §10A*/
```

Figure 14-1 Change prefix in EDGRMAIN REXX exec

2. Start the RMM ISPF dialog from the ISMF PRIMARY OPTION MENU by entering `R` to call the Removable Media Manager as shown in Figure 14-2. The selection available in the ISMF PRIMARY OPTION MENU depends on the “User Mode Selection” that can be changed by setting up your ISMF profile. Use the “0” option to change your profile, if necessary.

```
  Panel  Help
 ------------------------------------------------------------------------------
                  ISMF PRIMARY OPTION MENU - z/OS DFSMS V1 R10
Enter Selection or Command ===> R

Select one of the following options and press Enter:

0  ISMF Profile            - Change ISMF User Profile
1  Data Set                - Perform Functions Against Data Sets
2  Volume                  - Perform Functions Against Volumes
3  Management Class        - Specify Data Set Backup and Migration Criteria
4  Data Class              - Specify Data Set Allocation Parameters
5  Storage Class           - Specify Data Set Performance and Availability
9  Aggregate Group         - Specify Data Set Recovery Parameters
L  List                    - Perform Functions Against Saved ISMF Lists
R  Removable Media Manager - Perform Functions Against Removable Media
X  Exit                    - Terminate ISMF

Use HELP Command for Help; Use END Command to Exit.
```

Figure 14-2 Selecting the Removable Media Manager on the ISMF PRIMARY OPTION MENU

3. From the REMOVABLE MEDIA MANAGER (DFSMSrmm) ISPF primary menu, as shown in Figure 14-3 on page 576, type `5` to select the DFSMSrmm Commands menu. Type `REPORT` on the Option line to go directly to the primary DFSMSrmm Report Generator ISPF selection panel.

```
  Panel  Help
 ------------------------------------------------------------------------
                 REMOVABLE MEDIA MANAGER (DFSMSrmm) - z/OS V1R10
Option ===> 5

0  OPTIONS       - Specify dialog options and defaults
1  USER          - General user facilities
2  LIBRARIAN     - Librarian functions
3  ADMINISTRATOR - Administrator functions
4  SUPPORT       - System support facilities
5  COMMANDS      - Full DFSMSrmm structured dialog
6  LOCAL         - Installation defined dialog
X  EXIT          - Exit DFSMSrmm Dialog

Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-3 Selecting the COMMANDS option on the Removable Media Manager panel

4. Type R on the DFSMSrmm Command Menu, as shown in Figure 14-4, to get the primary DFSMSrmm Report Generator ISPF selection panel.

```
  Panel  Help
 ------------------------------------------------------------------------
                       DFSMSrmm Command Menu - z/OS V1R10
Option ===> R

0  OPTIONS   - Specify dialog options and defaults
1  VOLUME    - Volume commands
2  RACK      - Rack and bin commands
3  DATA SET  - Data set commands
4  OWNER     - Owner commands
5  PRODUCT   - Product commands
6  VRS       - Vital record specifications
7  CONTROL   - Display system control information
R  REPORT    - Report generator

Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-4 Selecting the Report generator option on the DFSMSrmm Command Menu panel

5. If you are using the DFSMSrmm Report Generator for the first time, you can set up your user options by entering 0 for Specify dialog options and defaults as shown in Figure 14-5 on page 577.

```
  Panel  Help
 ------------------------------------------------------------------------
                          DFSMSrmm Report Generator
Option ===> 0

0  OPTIONS         - Specify dialog options and defaults
1  REPORT          - Work with reports
2  REPORT TYPE     - Work with report types
3  REPORTING TOOL  - Work with reporting tools

Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-5 Selecting the Options option on the DFSMSrmm Report Generator panel

6. In the DFSMSrmm Dialog Options Menu panel, type option 1 to Specify processing options first as shown in Figure 14-6.

```
  Panel  Help
 ------------------------------------------------------------------------
                         DFSMSrmm Dialog Options Menu
Option ===> 1

1  USER      - Specify processing options
2  SORT      - Specify list sort options
3  REPORT    - Specify report options

Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-6 Selecting the User option on the DFSMSrmm Dialog Options Menu panel

7. Figure 14-7 on page 578 shows the DFSMSrmm Dialog User Options panel to specify the JOBCARDs to use for all jobs generated by DFSMSrmm.

```
  Panel  Help
 ------------------------------------------------------------------------
                          DFSMSrmm Dialog User Options
Command ===>

Date format . . . . . JULIAN          ( American, European, Iso or Julian )
Time zone . . . . . .       LOCAL     ( zone  offsetHH:MM:SS )

Confirm deletes . . . YES             ( Yes or No )

Processing option . . F                F - Foreground, B - Background

Eject option  . . . . C                C - Convenience, B - Bulk

Variable reuse  . . . Y                Y - Yes, N - No

Job statement information:-

 ===> //SCHLUMXX JOB ,RMM,NOTIFY=&SYSUID,
 ===> //         MSGCLASS=H,CLASS=O,MSGLEVEL=(1,1),REGION=0M
 ===> //*         ------ generated by DFSMSrmm ------
 ===> /*JOBPARM  SYSAFF=SC70

      Enter END command to save changes, or CANCEL to end without saving.
```

Figure 14-7 Specifying a JOBCARD on the DFSMSrmm Dialog User Options panel

8. After you return to the DFSMSrmm Dialog Options Menu panel, enter option 3 Specify report options, as shown in Figure 14-8.

```
  Panel  Help
 ------------------------------------------------------------------------
                          DFSMSrmm Dialog Options Menu
Option ===> 3

1  USER      - Specify processing options
2  SORT      - Specify list sort options
3  REPORT    - Specify report options

Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-8 Selecting the REPORT option on the DFSMSrmm Dialog Options Menu panel

9. In the DFSMSrmm Report Options panel, you specify the installation library that you want to use as your user library, as shown in Figure 14-9 on page 579. If you do not allocate the library, DFSMSrmm automatically allocates the library by using a primary space and secondary space of 10 tracks and 50 directory blocks.

```
  Panel  Help
 ------------------------------------------------------------------------
                            DFSMSrmm Report Options
Command ===>

Report definition libraries:
  User . . . . . . . . . . . 'MHLRES7.REPORT.LIB'
  Installation . . . . . . .
  Product  . . . . . . . . . 'SYS1.SAMPLIB'

User report JCL library  . . 'MHLRES7.REPORT.JCL'

DFSMSrmm allocates user libraries if they do not exist.
```

Figure 14-9 Selecting the report options on the DFSMSrmm Report Options panel

The following definitions refer to Figure 14-9:

Report definition libraries

This field enables you to specify or change the names of the libraries containing the report definitions, report types, and reporting tools.

The format of the report definition library members is fixed length, 80-byte records. You can use partitioned data sets or libraries. The DFSMSrmm product library is normally SYS1.SAMPLIB and contains report definitions and samples shipped with the product. Your installation defines the Installation library. You can name a user library. If you do not create the user library, DFSMSrmm automatically creates it for you. The information in the report definition libraries is used in a hierarchical order: first the user library, then the installation library, and finally the product library. When there are duplicate entries, the first occurrence in the hierarchy is used and others are ignored. Report type definitions are stored in the EDGGRTD member of each library and the reporting tool definitions are stored in the EDGGTOOL member.

Possible values are any data set name.

The default values are the library names that are defined in the EDGRMAIN exec:

User . . . . . 'MHLRES7.REPORT.LIB'
Installation . .
Product . . . . 'SYS1.SAMPLIB'

User report JCL library

This field enables you to specify or change the name of the library that contains the report JCL generated by DFSMSrmm. The format of the user report JCL library is fixed length, 80-byte records. You can use partitioned data sets or libraries.

The User report JCL library is required for you to generate and submit reports.

You must name a User report JCL library. If you do not create the User report JCL library, DFSMSrmm automatically creates it for you.

Possible values are any data set name.

The default value is the library name defined in the EDGRMAIN exec: 'MHLRES7.REPORT.JCL'

Return to the DFSMSrmm Dialog Options Menu by using one of the following options:

– Enter the `END` command to save changes.
– Enter the `CANCEL` command to end without saving.

Now that you have defined all required options, you can leave the DFSMSrmm Dialog Options Menu to return to the DFSMSrmm Report Generator primary option menu.

> **Note:** Set up the access lists for the libraries. Provide READ authority to the users of the installation libraries and the product libraries.

10. At the end, we will show you the currently supported tools that can be used to create reports. Figure 14-10 shows that ICETOOL and SYNCTOOL are the only tools that are supported. You can add, change, or delete a reporting tool at any time.

```
  Panel  Help
 ------------------------------------------------------------------------
                            DFSMSrmm Reporting Tools            Row 1 to 2 of 2
 Command ===>                                                  Scroll ===> PAGE

 The following line commands are valid: A,C, and D

 S Reporting tool                 Exec     Skeleton Colspace Group sort
 - ------------------------------ -------- -------- -------- ----------
   ICETOOL                        EDGRGGEN EDGSGICE     3        U
   SYNCTOOL                       EDGRGGEN EDGSGSYN     3        U
```

Figure 14-10 Reporting tool selection on the DFSMSrmm Reporting Tools panel

14.3 Create a report

Use the following the steps to create a report:

1. Select the Report Definition panel to customize report definitions that are shipped with the product. The *report definition* is a report file that contains all of the information that is needed to run a report. Each report definition in the product library, installation library, or user library contains the report type information, reporting tool information, the data fields that are used in the report, and the sort order of the records. The report selection criteria specify the subset of records that are used for a report. The reporting tool is a REXX EXEC that builds control statements to create reports that use a reporting utility, such as the DFSORT ICETOOL. You can change the reporting tool at any time.

   Type 1 from the DFSMSrmm Report Generator primary option menu as shown in Figure 14-11 to create a new report or work with existing reports.

```
 Panel  Help
 ------------------------------------------------------------------------------
                            DFSMSrmm Report Generator
 Option ===> 1

 0  OPTIONS         - Specify dialog options and defaults
 1  REPORT          - Work with reports
 2  REPORT TYPE     - Work with report types
 3  REPORTING TOOL  - Work with reporting tools

 Enter selected option or END command. For more info., enter HELP or PF1.
```

Figure 14-11 Selecting the REPORT option on the DFSMSrmm Report Generator menu panel

2. From the DFSMSrmm Report Definition Search panel that is shown in Figure 14-12, you can specify that all existing reports are listed or only reports generated by a specific user ID. You can also specify for which library to search. If you do not specify a selection, DFSMSrmm searches for all available reports in the user and product libraries when you press Enter.

.

```
 Panel  Help
 ------------------------------------------------------------------------------
                        DFSMSrmm Report Definition Search
 Command ===>

 Report name . .            May be generic.  Leave blank for all reports.

 User id . . . .            Leave blank for all user ids.

 Libraries (enter S):       Select one or more libraries.
   S  User                  Default is all defined libraries.
      Installation
   S  Product

 The following line commands will be available when the list is displayed:
   A - Add a new report definition  D - Delete a report definition
   G - Generate and save the JCL    J - Edit and manually submit the JCL
   L - List macro assembly results  M - Macros for report type are browsed
   N - Copy a report definition     S - Display or change the report definition
   T - Select a reporting tool
```

Figure 14-12 DFSMSrmm Report Definition Search panel

The following definitions refer to Figure 14-12 on page 581:

Report name

Use the Report name field to search for report definitions by name. The *report definition name* is the name of a member in one of the report definition libraries (user, installation, or product). The least generic report name that is used for the search in the product library is EDGG*. The following values are valid:

- One to eight alphanumeric or national characters for a specific report name.
- One to seven alphanumeric or national characters, followed by an asterisk (*), for a generic report name.
- An asterisk (*) for all report names. This is the default.
- The first character must be an alphabetic or national character.

User id

Use the User id field to specify the creation or last update user ID for report definitions. The following values are valid:

- One to eight alphanumeric or national characters for a specific user ID.
- An asterisk (*) for all user IDs. This is the default.
- The first character must be an alphabetic or national character.

Libraries

Select one or more of the user, installation, or product report definition libraries to specify which libraries are searched for report definitions. DFSMSrmm searches the libraries in the hierarchy shown so that duplicate report definition names in different libraries are skipped. If you add any members to the product library, use member names that start with the EDGG prefix.

The default is to search all defined libraries.

3. You get a list of all available reports shown in the DFSMSrmm Report Definitions panel in the form of a table as shown in Figure 14-13 on page 583. There are now 16 DFSMShsm related reports available. To show the details for the DCOLLECT BACKUP DATA report, type the `S` line command in front of this report.

```
  Panel  Help
 ------------------------------------------------------------------------------
                          DFSMSrmm Report Definitions         Row 1 to 23 of 40
Command ===>                                                   Scroll ===> PAGE
The following line commands are valid: A,D,G,J,L,M,N,S, and T

S Name     Report title                   Report type                User id
- -------- ------------------------------ ---------------------------- -------
  ARCGAB01 ABARS ABACKUP Statistics       DFSMShsm ABARS Report        HSM
  ARCGAR01 ABARS ARECOVER Statistics      DFSMShsm ABARS Report        HSM
S ARCGDB01 DCOLLECT BACKUP DATA           DFSMShsm DCOLLECT BACKUP     HSM
  ARCGDD01 DCOLLECT DASD CAPACITY PLANNIN DFSMShsm DCOLLECT DASD CAP   HSM
  ARCGDM01 DCOLLECT MIGRATION DATA        DFSMShsm DCOLLECT MIGRATION  HSM
  ARCGDT01 DCOLLECT TAPE CAPACITY PLANNIN DFSMShsm DCOLLECT TAPE CAP   HSM
  ARCGS001 Statistics for DFSMShsm        DFSMShsm FSR-SMF Records     HSM
  ARCGS002 Statistics for Backup          DFSMShsm FSR-SMF Records     HSM
  ARCGS003 Statistics for Migration       DFSMShsm FSR-SMF Records     HSM
  ARCGS004 Statistics for Recall          DFSMShsm FSR-SMF Records     HSM
  ARCGS005 Statistics for Recovery        DFSMShsm FSR-SMF Records     HSM
  ARCGS006 Statistics for Volume Dump     DFSMShsm FSR-SMF Records     HSM
  ARCGS007 Statistics for Restore from Du DFSMShsm FSR-SMF Records     HSM
  ARCGS008 Statistics for FRBACKUP        DFSMShsm FSR-SMF Records     HSM
  ARCGS009 Statistics for FRRecover       DFSMShsm FSR-SMF Records     HSM
  ARCGS010 DFSMShsm Thrashing Report      DFSMShsm FSR-SMF Records     HSM
  EDGGAUD1 SMF Audit of Volumes by Volser SMF Records for Volumes      RMM
  EDGGAUD2 SMF Audit of Volume by Rack    SMF Records for Volumes      RMM
  EDGGAUD3 SMF42 Audit of Volumes by Vols SMF42 Records for Volumes    RMM
  EDGGAUD4 SMF42 Audit of Volume by Rack  SMF42 Records for Volumes    RMM
  EDGGDCDS Data Sets by Storage Group     DFSMS DCOLLECT for Data Sets RMM
  EDGGDSNM Mixed Case data sets Retained  Extended Extract Records     DFRMM1
  EDGGREPL Volumes to be replaced         Extended Extract Records     RMM
  EDGGREPV Volumes to be replaced based o Extended Extract Records     RMM
  EDGGR01  Scratch tapes by volume serial Extended Extract Records     RMM
  EDGGR02  List of SCRATCH Volumes by Dat Extended Extract Records     RMM
  EDGGR03  Inventory List by Volume Seria Extended Extract Records     RMM
  EDGGR04  Inventory List by Dataset Name Extended Extract Records     RMM
  EDGGR06  Inventory of Volumes by Locati Extended Extract Records     RMM
  EDGGR07  Inventory of Dataset by Locati Extended Extract Records     RMM
  EDGGR08  Inventory of Bin by Location   Extended Extract Records     RMM
  EDGGR09  Datasets in Loan Location      Extended Extract Records     RMM
  EDGGR10  Volumes in Loan Location       Extended Extract Records     RMM
  EDGGR11  List MultiVolume and MultiFile Extended Extract Records     RMM
  EDGGR12  Movement Report by Dataset     Extended Extract Records     RMM
  EDGGR13  Movement Report by Bin         Extended Extract Records     RMM
  EDGGR14  Movement Report by Volume Seri Extended Extract Records     RMM
  EDGGR15  Volume Inventory Including Vol Extended Extract Records     RMM
  EDGGSEC1 Report of Accesses to Secure V SMF Security Records         RMM
  EDGGSEC2 SMF42 Report of Accesses to Se SMF42 Security Records       RMM
```

Figure 14-13 DFSMSrmm Report Definitions panel that shows all available default reports

> **Note:** You can also add new report types for data other than data that is created by DFSMSrmm. For example, the report types that are shipped with the report generator include types for DCOLLECT and DFSMShsm reporting. The *report type* contains information about a specific type of record in an input data set, the Assembler language macro that defines the record format, and basic record selection criteria. For example, the report type “Extract Records for Data Sets” in the product library contains information about the data set record in the extract data set, the EDGRDEXT mapping macro, and the minimum subset definition of records that are used in the report. Report types contain only the base information from which report definitions are created.

4. To modify an existing Report, type the `N` line command for a report definition line selection in front of the report you want to modify. In the pop-up panel, you must define a new name for this report as shown in Figure 14-14.

```
  Panel  Help
 ------------------------------------------------------------------------------
                          DFSMSrmm Report Definitions         Row 1 to 33 of 41
 Command ===>                                                  Scroll ===> PAGE

 The following line commands are valid: A,D,G,J,L,M,N,S, and T

 S Name     Report title                   Report type                  User id
 - -------- +-------------------------------------------------------+ ----- -------
   ARCGAB01 !                                                       !       HSM
   ARCGAR01 !                                                       !       HSM
 N ARCGDB01 !                                                       ! P     MHLRES7
   ARCGDD01 ! Enter the report name . . . . MYOWNREP                ! CAP   HSM
   ARCGDM01 !                                                       ! TION  HSM
   ARCGDT01 !                                                       ! CAP   HSM
   ARCGS001 +-------------------------------------------------------+ s     HSM
   ARCGS002 Statistics for Backup          DFSMShsm FSR-SMF Records     HSM
```

Figure 14-14 Specify a new report name

5. You can select one of the default reports or you can create your own report. To show how to create your own report, we show you an existing report and give you information that can be changed. We have selected the MYOWNREP DCOLLECT BACKUP DATA DFSMShsm DCOLLECT BACKUP report as shown in Figure 14-13 on page 583 to get the detailed information in the panel DFSMSrmm Report Definition - MYOWNREP as shown in Figure 14-15 on page 585.

```
  Panel  Help
 ------------------------------------------------------------------------------
                    DFSMSrmm Report Definition - MYOWNREP    Row 1 to 18 of 40
Command ===>                                                  Scroll ===> PAGE

Report title . . . DCOLLECT BACKUP DATA
Report footer  . .
Reporting tool . : ICETOOL                                 Report width:  260

Use END to save changes or CANCEL
Select a field name with S to specify a field selection criterion

S CO SO  Field name           Column header text                   CW  Len Typ
- -- --- -------------------- ------------------------------------ --- --- ---
*  1  1A UBDATE               LAST BU DATE                          12   4  N
   2  3A UBTIME               LAST BU TIME                          12   4  B
   3  2A UBDSNAM              DSN                                   44  44  C
   4     UBFRVOL              1st SRC VOL                           11   6  C
   5     UBDATCL              DC NAME                               30  30  C
   6     UBMGTCL              MC NAME                               30  30  C
   7     UBSTGCL              SC NAME                               30  30  C
   8     UBALLSP              ORIG ALLOC (KB)                       15   4  N
   9     UBUSESP              USER DATA (KB)                        14   4  N
  10     UBDSIZE              BACKUP DS (KB)                        14   4  N
*        DCURCTYP             RECORD TYPE FOR THIS RECORD           27   2  C
         DCUOUTH              DATA COLLECTION OUTPUT RECORD         29  24  C
...
```

Figure 14-15 DFSMSrmm Report Definition panel

The MYOWNREP report has 10 data columns and is sorted by date first, data set name, and time next.

The following column headings (see Figure 14-15) are defined:

S Selection column: Enter `S` in this selection column to list all fields with existing selection criteria, and to optionally add the newly selected field to the list to enable criteria to be entered. Fields for which a selection criterion already exists are marked with an asterisk in this column.

CO Column order: Use this data column to specify the order and the grouping of the columns in your report. The column order is left to right across the report page. Enter `G` to specify fields that you want to group. You must then specify the sort order for the fields within the group. Duplicate numbers are automatically resolved and renumbered from the bottom to the top of the list. Grouped field names appear on the top of the list by sort order. The grouped field names are the primary sort key of the record. Grouped field names are listed in the report header. A page break is forced each time that the contents of a grouping field name changes. *Grouped field names* are typically fields that have the same contents over several report pages, for example, the location and destination name in a movement report. Possible values are a decimal number between 1 - 99 or G.

SO Sort order: Use this data column to specify the sort order and the direction for the selected field names to change the sequence in which records are presented in the report.
Grouped field names are the primary sort key of the record. The sort order for grouped field names can only be specified within the range of the number of grouped field names. The default sort direction for grouped field names is ascending. Possible values are a decimal number between 1 - 99, followed by A - Ascending or D - Descending.

Field name The Field name data column displays the name of a field in the mapping macro that is specified in the report type definition.

Column header text Use the Column header text data column to specify the text used as the column header. The column header text, the column width, and the field length are dependent on each other. Changing the column header text sets the column width to the maximum of the column width, the field length, and the column header text. Changing the column width sets the length of the column header text to the column width. If the column width is less than the length of the field definition, the output in the report is truncated to the column width. Truncation removes trailing characters for character fields and leading characters for numeric fields. Possible values are 1 - 37 characters.

CW Column width: Use the column width data column to specify the length of the column in the report. The column header text, the column width, and the field length are dependent on each other. Changing the column header text sets the column width to the maximum of the column width, the field length, and the column header text. Changing the column width sets the length of the column header text to the column width. If the column width is less than the length of the field definition, the output in the report is truncated to the column width. Possible values are a decimal number between 1 - 99.

Len The field length data column displays the length of the field definition. If the column width is less than the length of the field definition, the output in the report is truncated to the column width.

Typ The field type data column displays the type of the field name. Use this information when specifying compare values. The following values are valid:

C - Character

N - Numeric

B - Bitstring

6. To add a new column to the report, you can type an `S` in the selection column and the position in which the column will be written in the report. In our example, we added the “DS size compressed in KB” information to the existing columns. This information will be shown as column 11 at the end of each row in the report. You can see our selection in Figure 14-16 on page 587.

```
  Panel  Help
 ------------------------------------------------------------------------------
                     DFSMSrmm Report Definition - MYOWNREP   Row 35 to 40 of 40
Command ===>                                                   Scroll ===> PAGE

Report title . . . DCOLLECT BACKUP DATA
Report footer  . .
Reporting tool . : ICETOOL                                  Report width:  260

Use END to save changes or CANCEL
Select a field name with S to specify a field selection criterion

S CO SO  Field name           Column header text                   CW  Len Typ
- -- --- -------------------- ------------------------------------ --- --- ---
         UBBKLNG              BLOCK LENGTH                          12   2  C
         UBFLAG2              INFORMATION FLAG #2                   19   1  B
         UBRECSP              RECOVER SPACE ESTIMATE(KB)            26   4  N
         UB_USER_DATASIZE     DS size if not compressed in KB       31   4  N
S 11     UB_COMP_DATASIZE     DS size compressed in KB              24   4  N
         UBBDSIE              END OF DCUBCDS                        14   1  C
```

Figure 14-16 DFSMSrmm Report Definition panel

7. Type `END` to save changes and return to the DFSMSrmm Report Definition panel or press Enter to tailor your selection DFSMSrmm Report Criteria panel as shown in Figure 14-17. If you have used END, go directly to step 15.

```
  Panel  Help
 ------------------------------------------------------------------------------
                      DFSMSrmm Report Criteria - MYOWNREP       Row 1 to 3 of 3
Command ===>                                                    Scroll ===> PAGE

Report title : DCOLLECT BACKUP DATA


Use END to save changes or CANCEL
The following line commands are valid: B,D,N,P,R,T, and I (for details)
Comparison operators: EQ =, NE <>, GT >, GE >=, LT <, LE <=, IN, BW, SE, SN
Conjunction: AND, OR, AND(, )AND

S Field name           Op Compare value(s)                        Conj Len Typ
- -------------------- -- ---------------------------------------- ---- --- ---
  DCURCTYP             EQ B                                            2  C
  UBDATE               LE &TODAY                                       4  N
- UB_COMP_DATASIZE     __ ________________________________________ ____  4  N
```

Figure 14-17 DFSMSrmm Report Criteria panel

The following column headings (see Figure 14-17) are defined:

S Use the S data column to specify the logical order of the criteria, or to request the Details panel. The following values are valid:

- **B** Bottom - Move this entry to the bottom.
- **D** Delete - Delete this entry.
- **I** Detail - Add or change information details, for example, substring parameters.
- **N** Next - Move this entry down by one.

P Previous - Move this entry up by one.
R Repeat - Repeat this entry.
T Top - Move this entry to the top.

8. To generate the job control for this report, return to the DFSMSrmm Report Definitions panel and type `G` in front of the report. You see the DFSMSrmm Report Generation panel shown in Figure 14-18 to change some variables before the file tailoring is run to create and store the job control.

```
                      DFSMSrmm Report Generation - MYOWNREP
Command ===>

Enter or change the skeleton variables for the generated JCL:

Input data set  . . . . 'MHLRES5.PEDCOL.EAVTST.G0003V00'

Date format . . . . . . YYYYDDD
    (American, European, Iso, Julian, or free form)
    Required if you use variable dates (&TODAY) in your selection criteria.

Create report data  . . N  (Y/N)
    Choose Y if you want an extract step included into your generated JCL.

Additional skeleton variables, if an extract step is included:
Skeleton Variable_1 . .
Skeleton Variable_2 . .
Skeleton Variable_3 . .
    The skeleton selection depends on the reporting macro . . . : ARCUTILP
                                          and macro keyword . . : TYPE=B
  Enter END command to start the report generation or CANCEL
```

Figure 14-18 DFSMSrmm Report Generation panel

The following fields (see Figure 14-18) are defined:

Input data set Specifies the name of the report input file. This field is optional for a report type definition, but required for the generation of the report JCL. If an extract step is included in the reporting JCL, the output file name of this step is also represented by this input data set parameter. In the case of an RMM extract step, the file must be pre-allocated. The value can be any sequential file name. Specify GDG data sets in the following way: 'RMM.EXTRACT(0)'

Date format Specifies the date format if you use variable dates with &TODAY... in the selection criteria. For RMM, extract data uses the date format that is specified when creating the data with the RMM report extract inventory management step. When you compare dates, remember that a format, such as ISO or JULIAN, enables easy comparison because of the year, month, and day hierarchy. The following values are valid:

AMERICAN Dates in format *MM/DD/YYYY*

EUROPEAN Dates in format *DD/MM/YYYY*

ISO Dates in format *YYYY/MM/DD*

JULIAN Dates in format *YYYY/DDD*

free form The free form has a maximum length of 20 bytes and contains *DD* and *MM* (alternatively *DDD*), and *YY* or *YYYY* or *CYY*. These key characters might be surrounded by fill-in characters, for example:

YYYYDDD This form fits for packed decimal data.

CYYDDD The C (century) is set to 1 for years after 2000.

DD.MM.YY The dot is used as a fill-in character.

' *YY*'*MM*'*DD*' Three leading blanks and ' as a fill-in character.

The default value JULIAN is specified in the DFSMSrmm Dialog User Option (option 0.1).

Create report data If you want to include the step that creates the report data, type `Y`. Additional skeleton variables might be needed for the extract JCL. The following values are valid:

N The Create Report Data step is not included.
Y The Create Report Data step is included.

Skeleton variables Use the skeleton variable fields to help with customization of the optional job step that creates the report data. Unless the skeleton variables are already defined in the report definition, the system tries to determine some skeleton variables based on the applied macro name. For example, if an RMM report extract macro is applied, the following occurs:

Skeleton Variable_1 Will contain DATEFORM(x).
Skeleton Variable_2 Will contain a suggestion for the RMM report extract message data set name.

Possible values are 1 - 50 characters.

9. Type `END` to save your changes and return to the DFSMSrmm Report Definitions panel. You get the information about where the JCL is stored as shown in Figure 14-19.

```
                          DFSMSrmm Report Definitions         Row 1 to 22 of 40
Command ===>                                                   Scroll ===> PAGE

The following line commands are valid: A,D,G,J,L,M,N,S, and T

S Name     Report title                   Report type                User id
- -------- ------------------------------ --------------------------- -------
  ARCGAB01 ABARS ABACKUP Statistics       DFSMShsm ABARS Report       HSM
  MYOWNREP DCOLLECT BACKUP DATA           DFSMShsm DCOLLECT BACKUP    MHLRES7
  ARCGDD01 DCOLLECT DASD CAPACITY PLANNIN DFSMShsm DCOLLECT DASD CAP  HSM
  ...
  EDGGAUD3 ———————————————————————————————————————————————— s    RMM
  EDGGAUD4 ! Report JCL MYOWNREP stored on 'MHLRES7.REPORT.JCL' ! s    RMM
  EDGGDCDS ———————————————————————————————————————————————— Sets RMM
  EDGGDSNM Mixed Case data sets Retained  Extended Extract Records    DFRMM1
....
```

Figure 14-19 DFSMSrmm Report Definitions panel with stored message

Now, you can see the stored report as member MYOWNREP in the library 'MHLRES7.REPORT.JCL' as shown in Figure 14-20 on page 590. This is the library that you defined in the Report definition libraries: USER as described in Figure 14-9 on page 579.

```
  Panel  Help
 ------------------------------------------------------------------------
EDIT              MHLRES7.REPORT.JCL                        Row 00001 of 00002
Command ===>                                                  Scroll ===> PAGE
           Name     Prompt       Size   Created          Changed          ID
_________ MYOWNREP                 89  2008/04/02  2008/04/02 07:20:08  MHLRES7
_________ EDGGSEC1                 89  2008/03/12  2008/03/12 21:26:16  MHLRES7
          **End**
```

Figure 14-20 User report definition library

10. Type the J (Edit and manually submit a report JCL line) command in front of the newly generated DCOLLECT BACKUP DATA report to edit the JCL member as shown in Example 14-1. You can see that the DS size compressed in the KB column is added to this report. Type SUBMIT and press Enter to submit the job.

Example 14-1 Sample DCOLLECT BACKUP DATA report

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT     MHLRES7.REPORT.JCL(MYOWNREP) - 01.00       Columns 00001 00072
Command ===>  SUBMIT                                   Scroll ===> PAGE
***************************** Top of Data ******************************
//RMMJOBS1 JOB (999,POK),MSGLEVEL=1,NOTIFY=&SYSUID
/*JOBPARM  SYSAFF=SC70
//*
//*******************************************************************
//**    SKELETON MEMBER EDGSGICE
//**    TAILORED BY THE RMM REPORT GENERATOR
//*******************************************************************
//**  DATE CALCULATION STEP
//**  COMPARE VALUES CONTAINING  ARE CALCULATED BASED ON RUN DATE
//**  COMPARE VALUES CONTAINING  ARE CALCULATED BASED ON RUN DATE
//DATECONV EXEC PGM=IKJEFT01,PARM='%EDGRGDAT'
//SYSPROC  DD DISP=SHR,DSN=SYS1.SEDGEXE1
//SYSTSPRT DD SYSOUT=*
//DATEFMT  DD *
DATE PATTERN:YYYYDDD
//INCLIN   DD *
 OPTION VLSHRT,VLSCMP
 INCLUDE COND=((9,2,CH,EQ,C'B'),
        AND,
               (85,4,PD,LE,&TODAY-000D))
 INREC FIELDS=(1,4,
           85,4,C' ',
           81,4,C' ',
           29,44,C' ',
           213,6,C' ',
           91,30,C' ',
           155,30,C' ',
           123,30,C' ',
           193,4,C' ',
           197,4,C' ',
           77,4,C' ',
           209,4,C' ')
 SORT FIELDS=(5,4,PD,A,
```

```
          15,44,CH,A,
          10,4,CH,A)
//INCLOUT  DD DSN=&INCL,UNIT=SYSALLDA,SPACE=(TRK,(1,1)),DISP=(,PASS),
//            DCB=(RECFM=FB,LRECL=80)
//SYSTSIN  DD DUMMY
//*******************************************************************
//**    following the new positions in record after inrec
//**    (variable record length needs a rdw field in column 1 to 4)
//*******************************************************************
//* UBDATE              : orig pos: 00081. position after inrec:     5
//* UBTIME              : orig pos: 00077. position after inrec:    10
//* UBDSNAM             : orig pos: 00025. position after inrec:    15
//* UBFRVOL             : orig pos: 00209. position after inrec:    60
//* UBDATCL             : orig pos: 00087. position after inrec:    67
//* UBMGTCL             : orig pos: 00151. position after inrec:    98
//* UBSTGCL             : orig pos: 00119. position after inrec:   129
//* UBALLSP             : orig pos: 00189. position after inrec:   160
//* UBUSESP             : orig pos: 00193. position after inrec:   165
//* UBDSIZE             : orig pos: 00073. position after inrec:   170
//* UB_COMP_DATASIZE    : orig pos: 00205. position after inrec:   175
//**
//WRITE1   EXEC PGM=ICETOOL,REGION=0M
//SYSPRINT DD SYSOUT=*
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//INDD     DD DSN=MHLRES5.PEDCOL.EAVTST.G0003V00,
//            DISP=SHR
//OUTDD    DD SYSOUT=*
//TEMP     DD UNIT=SYSALLDA,SPACE=(TRK,(5,25))
//TOOLIN   DD *
 SORT FROM(INDD)  TO(TEMP) USING(INCL)
 DISPLAY FROM(TEMP) LIST(OUTDD) -
 TITLE('DCOLLECT BACKUP DATA') -
 PAGE DATE(4MD/) TIME -
 HEADER('LAST BU DATE') ON(5,4,PD,A0) -
 HEADER('LAST BU TIME') ON(10,4,HEX) -
 HEADER('DSN') ON(15,44,CH) -
 HEADER('1st SRC VOL') ON(60,6,CH) -
 HEADER('DC NAME') ON(67,30,CH) -
 HEADER('MC NAME') ON(98,30,CH) -
 HEADER('SC NAME') ON(129,30,CH) -
 HEADER('ORIG ALLOC (KB)') ON(160,4,FI,A0) -
 HEADER('USER DATA (KB)') ON(165,4,FI,A0) -
 HEADER('BACKUP DS (KB)') ON(170,4,FI,A0) -
 HEADER('DS size compressed in KB') ON(175,4,FI,A0) -
 BLANK -
 TOTAL(' ')
//**                            /* &INCL CONTAINS THE SORT     §02A*/
//**                            /* INCLUDE STATEMENTS, MODIFIED §02A*/
//**                            /* IN STEP DATECONV            §02A*/
//INCLCNTL DD DSN=&INCL,DISP=(OLD,PASS)                      /*§02C*/
```

11. Submit and run the job to get the result as shown in Figure 14-21 on page 592. The column width can be changed in the DFSMSrmm Report Definition - MYOWNREP panel as described in step 4 in this section.

```
DCOLLECT BACKUP DATA        - 1 -        2008/04/03        05:15:10

    LAST BU DATE   LAST BU TIME   DSN
----------------   ------------   --------------------------------------------
         2007047   18054049       SYS1.SMS.ACDS.GW
         2007053   12395692       YYY.MHLRES2.MVOL.DATA
         2007053   12405017       YYY.SRCHFOR.LIST
         2007053   12405453       YYY.SUPERC.LIST
         2007054   18561271       MHLRES2.DCOLLECT.D987
         2007058   15290805       YYY.DCOLLECT.ESDS0
         2007058   15294019       YYY.DCOLLECT.ESDS01
         2007061   13334149       TESTFR.SUPERC.LIST
         2007071   17392521       TESTFR.CNTL.JCL
....
```

Figure 14-21 Sample DCOLLECT BACKUP DATA report

12. You can change your settings to reduce the column width and change the sequence in which the columns are shown in in the report. In the DFSMSrmm Report Definitions panel, type the S line command in front of the report to get the DFSMSrmm Report Definition - MYOWNREP detail panel. In Figure 14-22, we changed the column width of the LAST BU DATE, LAST BU TIME, and DS size compressed in KB. We also changed the column sequence, so we moved the DS size compressed in KB from column 11 to column 3 and added one to all columns between 3 and 10. That means that column 3 is now column 4, column 4 is now column 5, and so on.

```
                    DFSMSrmm Report Definition - MYOWNREP    Row 1 to 28 of 40
Command ===>                                                  Scroll ===> PAGE

Report title . . . DCOLLECT BACKUP DATA
Report footer  . .
Reporting tool . : ICETOOL                                  Report width:  271

Use END to save changes or CANCEL
Select a field name with S to specify a field selection criterion

S CO SO  Field name           Column header text                   CW  Len Typ
- -- --- -------------------- ------------------------------------ --- --- ---
*  1  1A UBDATE               LBU DATE                              8   4  N
   2  3A UBTIME               LBU time                              8   4  B
   3     UB_COMP_DATASIZE     DS size                               7   4  N
   4  2A UBDSNAM              DSN                                  44  44  C
   5     UBFRVOL              1st SRC VOL                          11   6  C
   6     UBDATCL              DC NAME                              30  30  C
   7     UBMGTCL              MC NAME                              30  30  C
   8     UBSTGCL              SC NAME                              30  30  C
   9     UBALLSP              ORIG ALLOC (KB)                      15   4  N
*        DCURCTYP             RECORD TYPE FOR THIS RECORD          27   2  C
...
```

Figure 14-22 Changed DFSMSrmm report definition for report MYOWNREP

Note: The column header text cannot exceed the column width that you specified in the Len column.

13. Type `END` to save your changes and return to the DFSMSrmm Report Definitions panel. Type `G` to generate and save the JCL, then, type `J` to edit and manually submit the JCL as described in steps 8 to step 1 in this section and submit the job. In Figure 14-23, you can see the changes. For example, the DS size column is now before the DSN column.

```
DCOLLECT BACKUP DATA        - 1 -        2008/04/03        06:08:07

        LBU DATE   LBU time            DS size   DSN
----------------   --------   ----------------   ------------------------------
         2007047   18054049                  0   SYS1.SMS.ACDS.GW
         2007053   12395692                  0   YYY.MHLRES2.MVOL.DATA
         2007053   12405017                  0   YYY.SRCHFOR.LIST
...
```

Figure 14-23 Changed DFSMSrmm report definition for report MYOWNREP

15

Splitting and merging your DFSMSrmm control data set

The control data set (CDS) contains information about the resources that are managed by Data Facility System Managed Storage removable media manager (DFSMSrmm). Normally, the resources are all those used on a single system or in a set of systems with a shared direct access storage device (DASD) environment. If you ever split these systems or merge them because of changes in workload, you might also want to do the same to the DFSMSrmm CDS.

This chapter describes these topics:

- Understanding when to split or merge a DFSMSrmm CDS
- Showing the process to successfully split or merge a DFSMSrmm CDS

By the end of this chapter, you will understand when to split or merge DFSMSrmm CDSs and the process to successfully accomplish the task.

15.1 When to merge or split

You typically do not have to merge or split the contents of the DFSMSrmm CDS. However, either during conversion or when you are changing the number of MVS systems that you run, you might want to merge or split the contents of the DFSMSrmm CDS.

15.1.1 During conversion

When you convert from an existing database to a DFSMSrmm CDS, you might want to use an existing DFSMSrmm CDS. You might have one system already converted to DFSMSrmm, and now a second system is to be converted, but you want to share the CDS with the first system.

During conversion, you add data produced by EDGCNVT to an existing CDS. You do not split the conversion data stream into multiple CDSs. See 15.5.1, “Converting to an existing CDS” on page 614 for details about how you merge the data you are converting into an existing DFSMSrmm CDS.

15.1.2 Changing the number of systems

If you change the number of systems or CDSs that you have and do not delete the information, merge or split the contents of the DFSMSrmm CDS:

- If you increase the number of CDSs, you split the CDS. For details about how you split an existing CDS into multiple DFSMSrmm CDSs, see 15.5, “Splitting the CDS” on page 609.
- If you decrease the number of CDSs, you merge CDSs. For details about how to merge multiple DFSMSrmm CDSs, see 15.4, “Merging the CDS” on page 605.

15.1.3 Considerations

When you merge or split a CDS, consider the timing of events and the contents of the CDS.

DFSMSrmm must be inactive, and you must not perform tape processing when you are processing the CDS. When you stop DFSMSrmm, back up existing CDSs and ensure that you have recovery plans in place in case you are unable to complete the process successfully. Also, when you stop DFSMSrmm, do not use the OPT=RESET function because it allows tapes to be processed without the knowledge of DFSMSrmm.

When merging or splitting a CDS, it is important to understand the contents of the CDS. For example, if you merge two CDSs, you cannot merge if the same resource is defined in each CDS. You must analyze the contents of the CDS to ensure that there are no duplicate resources. The following sections describe each resource type in the CDS and explain what you can do to handle duplicate resources.

Table 15-1 on page 597 lists the record types.

Table 15-1 Key types and record types

Key type	Record type
C	Control
CA	Action
D	Data set
E	Empty rack
F	Scratch rack
K	VRS
O	Owner
P	Product
R	Empty bin
S	In-use bin
U	In-use rack
V	Volume

15.1.4 Control record

Each DFSMSrmm CDS contains a control record. The control record contains the dates and times of when inventory management functions last ran, and the counts of rack numbers and bin numbers for built-in storage locations.

You can merge the control record from a single CDS and copy it to all target CDSs if you are splitting a CDS. In both cases, you must use EDGUTIL with PARM=MEND to correct the record counts after processing is completed.

The control record also contains the CDSID value. If you use this value to identify a CDS, you must ensure that each CDS that you create uses a different value. Use EDGUTIL with PARM='UPDATE' after processing is completed and specify the CDSID value, using the SYSIN DD CONTROL statement.

15.1.5 Action record

Action records contain information about the librarian actions that are outstanding. They include the release actions and any volume movements that are required. Omit these records, because there is a strong chance that they exist in each CDS, and the records are rebuilt by DFSMSrmm during inventory management.

15.1.6 Data set record

A data set record exists for each volume on which the data set is written. Each record is uniquely identified by the VOLSER and physical file number of the data set. When merging CDSs, see the information about volume records in 15.1.12, “Volume record” on page 602 for an explanation of how to handle duplicate volumes and the data sets on them.

When you split a CDS, you must copy all data set records to each target CDS, and then use the facilities of DFSMSrmm to delete those data set records that are not required. The DFSMSrmm EDGUTIL utility determines which data set records are required by using the volume information you have copied into each target CDS. Use the PARM=MEND option to delete the unnecessary data set records from the CDS.

15.1.7 VRS record

The vital record specification (*VRS) records* contain the retention and movement policies for your data. If you have multiple CDSs, it is likely that you have the same policies defined in each CDS for some types of data. The policies are identified by data set name and job name, VOLSER, and the VOLSER prefix.

When you merge CDSs, you must check for and resolve any duplicate policies. If you have a policy for the same data set name and job name combination in more than one CDS, you need to check that the retention and movement defined are the same. If they are the same, you can use the policy from any CDS. If they are not the same, you must resolve the conflict before merging the CDSs.

One way to resolve the conflict is to ensure that one CDS contains the complete set of VRS policies. Identify the CDS with most of the VRSs that you require in the merged CDS. Also, identify the VRSs that are unique in the other CDSs and define them in the one CDS.

When you split a CDS, you copy the complete set of VRS records to all target CDSs. This action might result in having VRSs that are unused in some CDSs, but it does not cause a problem in DFSMSrmm processing. When DFSMSrmm is started using the new CDS, you can use the DELETEVRS subcommand to delete the unused VRSs.

15.1.8 Owner record

There is one owner record for each identified user that owns a DFSMSrmm resource, such as a volume or VRS. There can also be owner records for users that own no resources. Perhaps, you have defined these owner records so that the information can be used within your installation.

The *owner record* contains detailed information about the user and the count of volumes that are owned. For users who own volumes, DFSMSrmm uses multiple owner records to provide directed retrieval of volume and data set information during search commands. You must consider the owner details and the owner volume records when splitting and merging the CDS.

Whether you merge or split CDSs, you must use a SORT program to select the owner records, remove the additional owner records that contain only volume information, and remove the duplicate owner records. After the records are loaded into a CDS, use the EDGUTIL utility with the PARM=MEND option to set the correct counts of owned volumes and build the owner volume records for owned volumes.

When you merge CDSs, check for duplicate owners across the CDSs that are being merged. Duplicates are not a problem because they can be handled as long as the owner record identifies the same person or task in each CDS. If the duplicate owners are the same user, determine which CDS has the correct owner details, such as address and contact information.

Ensure that at least one CDS has the correct owner details for the users, and use the owner details as the primary source of input to the merge. If the duplicate owners are really different users, you must decide how to handle that situation. You might have to assign different user IDs or define a new owner ID and transfer the owned resources to the new owner ID. Your objective is to have no duplicate owners, or to have any duplicates be for the same user in each case.

15.1.9 Product record

A *product (PP) record* identifies a single product and software level, and the volumes associated with that product. Whether you are merging or splitting a CDS, the existence of product records is a problem that must be resolved.

When you merge CDSs, there must be no duplicate product records. If there are, you must delete the duplicates and redefine the product information after the merge is completed. For each duplicate product that must be deleted, list the volumes for that product. To list the volumes, assuming that all product information is deleted from a single CDS, use the command that is shown in Figure 15-1.

```
RMM SEARCHPRODUCT NUMBER(*) LIMIT(*) NOLIST -
    CLIST('RMM LISTPRODUCT ')
EXEC EXEC.RMM LIST
```

Figure 15-1 RMM SEARCHPRODUCT subcommand

To delete all product information from a single CDS, enter the command shown in Figure 15-2.

```
RMM SEARCHPRODUCT NUMBER(*) LIMIT(*) NOLIST -
    CLIST('RMM DELETEPRODUCT ',' NORELEASE')
EXEC EXEC.RMM LIST
```

Figure 15-2 RMM SEARCHPRODUCT subcommand to delete all PP records

After you merge the CDSs, you can add product information by using the RMM CHANGEVOLUME subcommand in Figure 15-3.

```
RMM CHANGEVOLUME DZ11CO NUMBER(5695-DF1) FEATCD(A001) -
    LEVEL(V01R03M00)
```

Figure 15-3 RMM SEARCHPRODUCT subcommand to delete all PP records

When you split a CDS, the easiest way to handle product records is to target all product volumes to a single CDS. If this is not possible, you can choose either of the following actions:

- Delete the product information and add it after the CDS is split.
- Individually copy the product records by key to the correct target CDS. You can do this only if all volumes for a product are targeted to the same CDS.

15.1.10 Rack number record

There is a single rack record for each shelf location you have defined to DFSMSrmm. There must be at least one for each volume defined, but there can be more if you also defined empty rack numbers for use when volumes are added to the CDS. The *rack number* represents the volume's external VOLSER.

There are three rack number records, each using a unique key to aid retrieval by DFSMSrmm. During the copy operation, the three key ranges must be copied, but you need to consider only the rack numbers that you have defined.

When you merge CDSs, you must not have any duplicate rack numbers. Any duplicates must be deleted. If the duplicate already contains a volume, you can reassign the volume to another rack number by using the RMM CHANGEVOLUME subcommand. Figure 15-4 shows an example of two methods to reassign a volume to a new rack number.

```
RMM CHANGEVOLUME PP1234 POOL(A*)
RMM CHANGEVOLUME PP1234 RACK(A00001)
```

Figure 15-4 RMM CHANGEVOLUME subcommand

In the first command, DFSMSrmm picks an empty rack in pool A*. In the second example, you select the empty rack to be used.

Before you clean up any duplicate rack numbers, see 15.1.12, “Volume record” on page 602 for an explanation of how to handle duplicate volumes.

When you split a CDS, you split the rack numbers in the same way that you split the volume ranges into the target CDSs. You decide which volumes are copied to each CDS, and ensure that the rack numbers with which they are associated are also copied to the same target CDS.

During the copy process, you identify the rack numbers using key ranges to include or exclude the appropriate rack numbers. The significant part of the key of rack number records is the first 16 characters. The key must be specified in hexadecimal, because the second byte is hex zero.

When copying ranges of volumes, you can use the COUNT keyword on the REPRO statement. However, you must specify a hexadecimal key for the TOKEY and the FROMKEY. The key includes an 8-character media name that matches the VLPOOL media name and the rack number. Example 15-1 shows an example of copying the rack numbers for the volumes in the 001000 - 001999 range.

Example 15-1 Copying rack numbers

```
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
      FROMKEY(X'C500F3F4F8F040404040F0F0F1F0') -
        TOKEY(X'C500F3F4F8F040404040F0F0F1F9F9F9')
      /*  COPY racks from E 3480    001000       to  E  3480    001999 */
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
      FROMKEY(X'C600F3F4F8F040404040F0F0F1F0') -
        TOKEY(X'C600F3F4F8F040404040F0F0F1F9F9F9')
      /*  COPY racks from F 3480    001000       to  F  3480    001999 */
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
      FROMKEY(X'E400F3F4F8F040404040F0F0F1F0') -
        TOKEY(X'E400F3F4F8F040404040F0F0F1F9F9F9')
      /*  COPY racks from U 3480    001000       to  U  3480    001999 */
```

The first character of the key is either E, F, or U; the media name starts in position 3; and the rack number starts in position 11. Only copy a range of volumes to a single target control data set.

15.1.11 Bin number record

There is a single *bin record* for each storage shelf location you have defined to DFSMSrmm.

There are two different bin number records, each using a unique key to aid retrieval by DFSMSrmm. During the copy operation, the two key ranges must be copied, but you only need to consider the bin numbers that you have defined.

When you merge CDSs, you must not have any duplicate bin numbers. Any duplicates must be deleted. If the duplicate already contains a volume, you can reassign the volume to another bin number by using the RMM CHANGEVOLUME subcommand. You can reassign a volume to a new bin number by typing the command in Figure 15-5.

```
RMM CHANGEVOLUME A00123 BIN(000029)
```

Figure 15-5 RMM CHANGEVOLUME subcommand

You can also remove the volume from the storage location, free up the bin number, and delete the unused bin number. Use the following command to delete duplicate bin number X00022 in location VAULT1 of media name CARTS that you see in Figure 15-6.

```
RMM CHANGEVOLUME A00123 LOCATION(HOME) CMOVE
RMM DELETEBIN X00022 MEDIANAME(CARTS) LOCATION(VAULT1)
```

Figure 15-6 RMM CHANGEVOLUME subcommand to confirm outstanding moves

The next time that you run inventory management, the volume is reassigned to the storage location, and another bin number is assigned.

Note: When you need to reassign bin numbers, consider how offsite movements are managed in your installation. If you depend on the volume being in a specific slot in a storage location, ensure that any volumes you change are physically moved in the storage location.

When you split a CDS, you split the bin numbers according to how you want to handle storage locations in the split environment. If the volumes in the storage locations are split so that each target CDS contains volumes for each shelf-managed storage location, handling the splitting of bin numbers is complex. We suggest that you use DFSMSrmm commands to change the locations of all volumes and confirm their return to their home locations. This approach frees the in-use bin numbers.

Do not copy the bin numbers to the target CDSs. Instead, add empty bin numbers after the split and use inventory management VRSEL and DSTORE processing to assign new bin numbers to the volumes after the split processing is completed.

Targeting all volumes in a single storage location for the same CDS simplifies the copying of bin numbers. During the copy process, you identify the bin numbers using key ranges to include or exclude the appropriate bin numbers. The significant part of the key of bin number records is the first 24 characters. The key can be specified in character form.

When copying ranges of bin numbers, specify a TOKEY and the FROMKEY because a range of bin numbers can include both empty and in-use bin numbers. The key includes an 8-character media name, which matches the LOCDEF media name as well as the bin number and the location name. Example 15-2 shows an example of copying the bin numbers for the bin numbers in storage location VLT1, media name CARTS, and the V10001 - V10500 range.

Example 15-2 Sample REPRO statements for copying bin numbers

```
REPRO INFILE(SOURCE) OUTFILE(OUTS1)         -
      FROMKEY(C'RUVLT1    CARTS   V10001') -
        TOKEY(C'RUVLT1    CARTS   V10500')
REPRO INFILE(SOURCE) OUTFILE(OUTS1)         -
      FROMKEY(C'SUVLT1    CARTS   V10001') -
         TOKEY(C'SUVLT1    CARTS   V10500')
```

The first two characters of the key are either RU or SU; the storage location is in position 3; the media name starts in position 11; and the bin number starts in position 19. Only copy a range of bin numbers to a single target CDS.

15.1.12 Volume record

The *volume record* contains detailed information about a volume and its contents. It also identifies the data sets that are on the volume and volume chaining information. Some fields in the records identify the existence of other records in the CDS.

When moving volume records from one CDS to another, you must be sure to move the related records, as well. These include the data set records, owner record, rack record, bin records, and product record. If you do not move them, the EDGUTIL utility can identify the missing records, and, with the MEND option, creates the missing records with one exception, which is the product records that cannot be re-created.

When you merge CDSs, you must not have any duplicate volumes. If you do, you must either skip the duplicate volumes or delete them from the source CDS before starting the merge process. You need to understand why there are duplicate volumes, and based on your understanding, select the best approach for dealing with them.

If you have duplicate volumes, there can also be duplicate data set records for those volumes. If you delete the volumes, you also delete the data set records.

When you split a CDS, you most likely split the volumes according to VOLSER ranges. During the copy process, you identify the volumes using key ranges to include or exclude the appropriate volumes. The significant part of the key of volume records is the first eight characters. The key must be specified in hexadecimal because the second byte is hex zero. When copying ranges of volumes, you can use the COUNT keyword on the REPRO statement. Otherwise, you must specify a hexadecimal key for the TOKEY, as well.

Example 15-3 shows REPRO statements for copying the volumes in the 001000 - 001999 range.

Example 15-3 Sample REPRO statements for copying ranges of volumes

```
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'E500F0F0F1') COUNT(1000)
```

The first character of the key is V, and the VOLSER starts in position 3. Only copy a range of volumes to a single target CDS.

15.2 PARMLIB options

The following PARMLIB options are affected by splitting and merging CDSs:

- OPTION
- LOCDEF
- VLPOOL
- REJECT
- OPENRULE
- PRTITION

15.2.1 OPTION

In the OPTION command there are several commands should be checked.

CDSID

When you start DFSMSrmm, the CDSID ID is compared to the ID in the control data set control record. If the IDs match, DFSMSrmm startup continues. If the control data set does not have an ID, DFSMSrmm creates the ID in the control record from the CDSID. If the IDs do not match, DFSMSrmm startup fails and DFSMSrmm issues a message to the operator to select another parmlib member.

CATSYSID

Specify CATSYSID to enable DFSMSrmm catalog synchronization. Use the CATSYSID operand to identify the user catalogs you want tracked when you run the DFSMSrmm EDGHSKP utility with the CATSYNCH operand to exploit catalog status tracking.

DSNAME

Specifies the name of the DFSMSrmm control data set.

JRNLNAME

Specifies the name of the journal.

15.2.2 LOCDEF

These options identify your locations to DFSMSrmm. When you are merging CDSs, consolidate the LOCDEF options from all CDSs you are merging to ensure that you have no duplicates. If you have duplicates, ensure that they are identified as the same location type, and with the same media names.

When you split a CDS, each system that will use one of the target CDSs must have defined to it the locations where the volumes in the target CDS reside. You can simply include all LOCDEFs in each new system, but you have the option to remove those that are not required.

15.2.3 VLPOOL

These options identify the ranges of rack numbers and pools of shelf space that you have in your library. If your library is not changing, we suggest that you do not change your current VLPOOL options.

If you merge CDSs and your existing VLPOOL definitions are different for each existing control data set, you must merge the definitions together to cover the merged sets of volumes and rack numbers. Ensure that the operands specified on each VLPOOL are correct when the same pool prefix is defined in more than one source environment.

When you split a CDS, we suggest that you take the existing VLPOOL definitions and use them on each of the new CDSs.

15.2.4 REJECT

These options identify the ranges of volumes to be rejected for use on the system. The REJECT specifies the rack number prefix and needs to match the VLPOOL definitions that you have.

Where you have a tape library that is shared by multiple systems, define the same VLPOOL definitions to all systems. Then, use the REJECT definitions to prevent systems from using volumes that are for use on other systems.

The actions that you take for the REJECT definitions need to match how you handle the VLPOOL definitions.

Important: If you currently use REJECT commands, you have to convert from the use of REJECT commands in order to use the PRTITION and OPENRULE commands.

15.2.5 OPENRULE

These options identify special processing for DFSMSrmm to perform during OPEN processing. They apply equally to both system-managed and non-system-managed volumes.

In a shared CDS RMMplex, these commands allow you to control sharing volumes between systems while still maintaining a single CDS. OPENRULE allows you to specify what cannot be used as well as what can be used.

15.2.6 PRTITION

These options identify special processing for DFSMSrmm to perform during library entry, export/import, eject, Common User Access (CUA), OPEN, and EXPROC processing. The PRTITION commands apply to all volumes, not only system-managed volumes. However, only system-managed volumes benefit from the functions, such as library insert, CUA, and eject.

15.3 Running IDCAMS REPRO

When using IDCAMS REPRO to copy records to the target CDS, you might receive messages indicating that records are duplicates, or that there are no input records:

- Return code 4 from a REPRO command with this message:

  ```
  IDC11465I DELIMITERS WERE SPECIFIED, BUT NO RECORDS PROCESSED.
  ```

  The only key range for which this message occurs is P for product records. Expect this message if you do not have any product definitions in the CDS.

- Return code 8 or 12 from a REPRO command with messages:

  ```
  IDC3302I  ACTION ERROR ON DFRMM.MASTER.MERGE
  IDC3308I ** DUPLICATE RECORD X'ccXXXXXXXX'
  ```

  You do not receive these messages if you follow the suggestions made earlier. After four duplicate records, IDCAMS stops processing the records, so the copy operation is incomplete. You must resolve the duplicates by taking the actions that are suggested in 15.1.3, “Considerations” on page 596, and then rerun the merge or split job.

15.4 Merging the CDS

Use the following steps to merge two CDSs. SYS1 is the system on which the merge is being performed, and SYS2 is the second system from which the CDS data is being merged.

1. See 15.1.3, “Considerations” on page 596, and use the information to analyze the CDSs that are being merged. If any potential problems or duplicate records are identified, make the suggested changes before continuing.
2. Stop DFSMSrmm on all systems affected by merging the CDSs. You cannot use tapes while you merge CDSs.
3. Back up the control data set on SYS2 and send the backup to SYS1. Run the backup job shown in Example 15-4 on page 606. If you have shared DASD between SYS1 and SYS2, you can skip this step.
4. Restore the SYS2 CDS on SYS1. Run the restore job shown in Example 15-5 on page 606. If you have shared DASD between SYS1 and SYS2, you can skip this step.
5. Run the merge job shown in Example 15-6 on page 607, including these steps:
   a. BACKS1: Backs up the CDS used for SYS1.
   b. COPYO: Copies the owner records from both source CDSs to temporary files for input to step SORTO.
   c. SORTO: Removes any duplicate owner records found in SYS2 input and deletes the owner volume records.
   d. DEFINE: Deletes and then redefines a target CDS.
   e. COPY: Copies most of the records from both source CDSs. For VRS and product records, only one CDS is used as input, and a later step copies from the second CDS. This is done because these are the most likely records to still have duplicates.
   f. COPYK2: Copies the VRS records from SYS2. Expect duplicates to be found if you use the same VRS definitions on both CDSs. You can remove this step if you know that all VRS definitions were in SYS1.
   g. COPYP2: Copies the product records from SYS2.

h. MEND: Validates the newly merged CDS and makes any corrections to record relationships. It also builds the owner volume records, corrects the count of owned volumes, and corrects the rack number and bin number counts in the control record.

i. VERIFY: Verifies that the CDS is logically sound and ready to use. If the merge job is successful, you are ready to use the merged control data set.

6. Update SYS1.PARMLIB member EDGRMMxx as indicated in Table 15-2.

Table 15-2 Commands for changing EDGRMMxx

Command	Required change
OPTION	Update DSNAME to identify the new CDS name.
LOCDEF	Update to reflect all locations used in the merged CDS.
VLPOOL	Update to reflect the new set of pools.
REJECT	Update to reflect any changes in VLPOOL definitions.
OPENRULE	Update to specify the use of volumes.
PRTITION	Update to reflect any changes in VLPOOL definitions.

7. Start the DFSMSrmm procedure using the new PARMLIB member.
8. Check that the merged information is good by using DFSMSrmm commands, and the ISPF dialog to retrieve and display information.
9. Run any commands that are identified to be run in 15.1, “When to merge or split” on page 596.
10. Run DFSMSrmm inventory management on the new CDS. During this run, DFSMSrmm VRSEL and DSTORE processing identifies volumes to be moved to the correct location. If any moves are identified, use EDGRPTD to produce the picking lists and actions of any moves that you know are still required. The inventory management run typically executes all inventory management functions, including backup.

Example 15-4 shows the JCL for backing up the CDS on SYS2.

Example 15-4 JCL to back up the CDS on SYS2

```
//BACKS2    EXEC PGM=EDGBKUP,PARM='BACKUP'
//SYSPRINT  DD   SYSOUT=*
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//BACKUP    DD   DSN=DFRMM.MASTER.BACKUP2(+1),DISP=(,CATLG,DELETE),
//         UNIT=SYSDA,SPACE=(9216,(1000,999),RLSE),
//         LRECL=9216,DSORG=PS,BLKSIZE=0,RECFM=VB
```

Example 15-5 shows the JCL for restoring the SYS2 CDS on SYS1.

Example 15-5 JCL to restore the SYS2 CDS on SYS1

```
//RESTS2    EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//BACKUP    DD   DSN=DFRMM.MASTER.BACKUP2(0),DISP=SHR
```

Example 15-6 shows the JCL for executing the merge process for two CDSs.

Example 15-6 Merge JCL

```
//BACKS1   EXEC PGM=EDGBKUP,PARM='BACKUP'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DSN=DFRMM.MASTER.SYS1,DISP=SHR
//BACKUP   DD   DSN=DFRMM.MASTER.BACKUP1(+1),DISP=(,CATLG,DELETE),
//         UNIT=SYSDA,SPACE=(6233,(1000,999),RLSE),
//         DCB=(MODEL.VB9216,DSORG=PS)
//*****************************************************************
//** COPY THE OWNER RECORDS FROM BOTH SOURCE CDSs TO             **
//** A SEQUENTIAL DATASET TO BE USED FOR THE MERGED CDS.         **
//*****************************************************************
//COPY0    EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//INS1     DD   DSN=DFRMM.MASTER.SYS1,DISP=SHR
//INS2     DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//OUTS1    DD   DSN=&&INO1,DISP=(,PASS),UNIT=SYSDA,
//         SPACE=(1024,(5000,5000),RLSE),
//         LRECL=2048,RECFM=VB
//OUTS2    DD   DSN=&&INO2,DISP=(,PASS),UNIT=SYSDA,
//         SPACE=(1024,(5000,5000),RLSE),
//         LRECL=2048,RECFM=VB
//SYSIN    DD   *
 REPRO INFILE(INS1) OUTFILE(OUTS1) FROMKEY(0) TOKEY(0)
 REPRO INFILE(INS2) OUTFILE(OUTS2) FROMKEY(0) TOKEY(0)
//*****************************************************************
//** SORT THE OWNER  RECORDS FROM BOTH SYSTEMS TO A TEMPORARY    **
//** SEQUENTIAL DATASET TO BE PASSED TO A LATER STEP COPYALL.    **
//** REMOVE DUPLICATE OWNERS FROM SOURCE S2.                     **
//*****************************************************************
//SORT0    EXEC PGM=SORT,REGION=6M
//SYSPRINT DD   SYSOUT=*
//SORTIN   DD   DSN=&&INO1,DISP=(OLD,DELETE)
//         DD   DSN=&&INO2,DISP=(OLD,DELETE)
//SORTWK01 DD   UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTOUT  DD   DISP=(NEW,PASS),UNIT=SYSDA,DCB=*.SORTIN,
//         SPACE=(1024,(5000,5000),RLSE),DSN=&&SORT0
//SYSOUT   DD   SYSOUT=*
//SYSIN    DD   *
 SORT FIELDS=(5,9,CH,A)
 INCLUDE COND=(5,1,CH,EQ,C'O',AND,14,1,CH,EQ,X'00')   BASE OWNERS ONLY
 SUM  FIELDS=NONE                                       ELIMINATE DUPS
//DEFINE   EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//MSTVOL   DD  DISP=SHR,UNIT=3380,VOL=SER=VSMN01
//SYSIN    DD  *
  DEL DFRMM.MASTER.MERGE
  SET MAXCC=0
  DEF CL(NAME(DFRMM.MASTER.MERGE) FILE(MSTVOL)             -
         VOLUME(VSMN01) KILOBYTES(50000 9000) FSPC(20 20) -
         RECORDSIZE(512 9216)  KEYS(56 0) SHR(3 3) RUS)  -
      DATA(NAME(DFRMM.MASTER.MERGE.DATA)            -
           CISZ(10240) )                -
      INDEX(NAME(DFRMM.MASTER.MERGE.INDEX) IMBED REPL     -
           CISZ(2048) )
//*****************************************************************
//**              TARGET SYSTEM                                  **
//** COPY THE RECORDS FROM BOTH SOURCE CDS TO THE TARGET CDS     **
//*****************************************************************
//COPY     EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
```

```
//INS1     DD   DSN=DFRMM.MASTER.SYS1,DISP=SHR
//INS2     DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//TARGET   DD   DSN=DFRMM.MASTER.MERGE,DISP=SHR
//OWNER    DD   DSN=&&SORTO,DISP=(OLD,DELETE)
//SYSIN    DD   *
 REPRO INFILE(INS1) OUTFILE(TARGET) FROMKEY(C) TOKEY(C) COUNT(1)
 REPRO INFILE(INS1) OUTFILE(TARGET) FROMKEY(D) TOKEY(F)
 REPRO INFILE(INS2) OUTFILE(TARGET) FROMKEY(D) TOKEY(F)
 REPRO INFILE(INS1) OUTFILE(TARGET) FROMKEY(K) TOKEY(K)
 REPRO INFILE(OWNER) OUTFILE(TARGET)
 REPRO INFILE(INS1) OUTFILE(TARGET) FROMKEY(P) TOKEY(P)
 REPRO INFILE(INS1) OUTFILE(TARGET) FROMKEY(R) TOKEY(V)
 REPRO INFILE(INS2) OUTFILE(TARGET) FROMKEY(R) TOKEY(V)
//*****************************************************************
//**              SOURCE CDS 2 - ON TARGET SYSTEM               **
//** COPY "K" RECORDS FROM THE SOURCE CDS 2 TO THE TARGET CDS.  **
//** RETURN CODE 8 or 12 INDICATES DUPLICATE RECORDS.           **
//*****************************************************************
//COPYK2   EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//INS2     DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//TARGET   DD   DSN=DFRMM.MASTER.MERGE,DISP=SHR
//SYSIN    DD   *
 REPRO INFILE(INS2) OUTFILE(TARGET) FROMKEY(K) TOKEY(K)
//*****************************************************************
//**              SOURCE CDS 2 - ON TARGET SYSTEM               **
//** COPY "P" RECORDS FROM THE SOURCE CDS 2 TO THE TARGET CDS.  **
//** RETURN CODE 8 or 12 INDICATES DUPLICATE RECORDS.           **
//*****************************************************************
//COPYP2   EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//INS2     DD   DSN=DFRMM.MASTER.SYS2,DISP=SHR
//TARGET   DD   DSN=DFRMM.MASTER.MERGE,DISP=SHR
//SYSIN    DD   *
 REPRO INFILE(INS2) OUTFILE(TARGET) FROMKEY(P) TOKEY(P)
//*****************************************************************
//**              MERGED TARGET CDS                             **
//** RUN EDGUTIL WITH   PARM='MEND' TO UPDATE THE NEW MERGED    **
//**   CDS TO HAVE THE CORRECT COUNTS FOR RACK/BINS.            **
//**   ALSO SETS OWNER VOLUME COUNTS AND BUILDS OWNER VOLUME    **
//**   INFORMATION.                                             **
//*****************************************************************
//MEND     EXEC PGM=EDGUTIL,PARM='MEND'
//MASTER   DD   DISP=SHR,DSN=DFRMM.MASTER.MERGE
//SYSPRINT DD   SYSOUT=*
//*****************************************************************
//**              MERGED TARGET CDS                             **
//** RUN EDGUTIL WITH   PARM='VERIFY(ALL)' ON THE NEW MERGED    **
//**   CDS TO VERIFY THE INTEGRITY OF THE NEW CDS.              **
//*****************************************************************
//VERIFY   EXEC PGM=EDGUTIL,PARM='VERIFY(ALL)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   SUBSYS=(BLSR,'DDNAME=MASTERB')
//MASTERB  DD   DISP=SHR,DSN=DFRMM.MASTER.MERGE
//* Use of Batch LSR can speed the running of EDGUTIL
//* IF you have not got BLSR setup on your system you must replace
//* the MASTER DD statement with the following one, and delete
//* the MASTERB DD statement
```

```
//*MASTER  DD  DISP=OLD,DSN=DFRMM.MASTER.MERGE,
//*             AMP=('BUFND=50,BUFNI=50')
//SYSIN    DD  DUMMY
```

15.5 Splitting the CDS

Use the following steps to split a CDS into two CDSs. SYS1 and SYS2 are the new systems to which CDS data is being split:

1. See 15.1.3, “Considerations” on page 596, and use the information to analyze the CDS that is being split. If you identify any potential problems, make the suggested changes before continuing. Consider how you plan to split the ranges of volumes, rack numbers, and bin numbers. You must prepare the access method services (AMS) statements for use in job steps COPYV and COPYB.
2. Stop DFSMSrmm on all systems affected by splitting the CDS. You cannot use tapes while you split a CDS.
3. Run the merge job that is shown in Example 15-6 on page 607. The job includes the following steps:
   a. BACKS: Backs up the CDS to be split.
   b. COPYO: Copies the owner records from the source CDS to a temporary file for input to step SORTO.
   c. SORTO: Removes the owner volume records.
   d. DEFINE: Deletes and then redefines the target CDSs.
   e. COPY: Copies those records from the source CDS that must be copied to both target CDSs. Copying the record ranges that need to be split is performed in later steps.
   f. COPYP2: Copies the product records to SYS2. This assumes that all product volumes are split out to the SYS2 CDS. If this is not the case, see “Product record” on page 620 and update the JCL based on the suggested actions.
   g. COPYV: Copies the volume and rack numbers to the two target CDSs. The JCL example shows how you can specify to AMS which records are to be copied. You must update the sample AMS statements to copy the ranges that you want to have in each target CDS.
   h. COPYB: Copies the bin numbers to the two target CDSs. The JCL example shows how you can specify to AMS which records to copy. You must update the sample AMS statements to copy the actual ranges that you want to have in each target CDS.
   i. MEND1: Validates the newly created SYS1 CDS and makes any corrections to record relationships. It also builds the owner volume records, corrects the count of owned volumes, corrects the rack number and bin number counts in the control record, and deletes any unused data set records.
   j. MEND2: Validates the newly created SYS2 CDS and makes any corrections to record relationships. It also builds the owner volume records, corrects the count of owned volumes, corrects the rack number and bin number counts in the control record, and deletes any unused data set records.
   k. VERIFY1: Verifies that the SYS1 CDS is logically sound and ready to use.
   l. VERIFY2: Verifies that the SYS2 CDS is logically sound and ready to use.
   m. If the merge job is successful, you are ready to use the new CDSs.

4. Back up the newly created CDSs for SYS1 and SYS2. Run the backup job that is shown in Example 15-8 on page 614.
5. Update SYS1.PARMLIB member EDGRMMxx on all affected systems as shown in Table 15-3.

Table 15-3 Commands to update EDGRMMxx

Command	Required change
OPTION	Update DSNAME to identify the new CDS name and journal data set name. Update SYSID to identify a unique system ID. If you do not specify a value, DFSMSrmm uses the System Management Facilities (SMF) system ID. If you specified CDSID in PARMLIB for the source CDS, ensure that CDSID still matches the value you used on the source CDS.
OPENRULE	Update to control use of volumes.
PRTITION	Update to reflect any changes in VLPOOL definitions.
LOCDEF	Update to reflect the locations used in each CDS.
VLPOOL	Update to reflect the new set of pools. Update the SYSID to match the value specified on the OPTION SYSID keyword. If SYSID is not specified, ensure that the VLPOOL SYSID, if required, matches the SMF system ID of the running system.
REJECT	Update to reflect any changes in VLPOOL definitions.

6. Start the DFSMSrmm procedure on each system by using the new PARMLIB member.
7. Check that the merged information is good by using DFSMSrmm commands and the ISPF dialog to retrieve and display information.
8. Execute any commands that are identified to be run in 15.1, “When to merge or split” on page 596.
9. Run DFSMSrmm inventory management on each of the new CDSs. During this run, DFSMSrmm, VRSEL, and DSTORE processing identifies volumes to be moved to the correct location. If any moves are identified, use EDGRPTD to produce the picking lists and actions of any moves that you know are still required. The inventory management run needs to execute all inventory management functions, including backup.

Example 15-7 shows the JCL for executing the split process to create two target CDSs.

Example 15-7 Sample JCL to split a DFSMSrmm CDS

```
//BACKS     EXEC PGM=EDGBKUP,PARM='BACKUP'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DSN=DFRMM.MASTER.CDS,DISP=SHR
//BACKUP    DD   DSN=DFRMM.MASTER.BACKUP(+1),DISP=(,CATLG,DELETE),
//          UNIT=SYSDA,SPACE=(6233,(1000,999),RLSE),
//          DCB=(MODEL.VB9216,DSORG=PS)
//*****************************************************************
//** COPY THE OWNER RECORDS FROM SOURCE CDS TO                   **
//** A SEQUENTIAL DATASET TO BE USED FOR THE NEW CDSs            **
//*****************************************************************
//COPYO    EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//SOURCE   DD   DSN=DFRMM.MASTER.CDS,DISP=SHR
//OUT      DD   DSN=&&OUTO,DISP=(,PASS),UNIT=SYSDA,
//         SPACE=(1024,(5000,5000),RLSE),
//         LRECL=2048,RECFM=VB
//SYSIN    DD    *
```

```
 REPRO INFILE(SOURCE) OUTFILE(OUT) FROMKEY(0) TOKEY(0)
//*****************************************************************
//** SORT THE OWNER  RECORDS FROM BOTH SYSTEMS TO A TEMPORARY   **
//** SEQUENTIAL DATASET TO BE PASSED TO A LATER STEP COPYALL.   **
//** REMOVE DUPLICATE OWNERS FROM SOURCE S2.                    **
//*****************************************************************
//SORTO    EXEC PGM=SORT,REGION=6M
//SYSPRINT DD   SYSOUT=*
//SORTIN   DD   DSN=&&OUTO,DISP=(OLD,DELETE)
//SORTWK01 DD   UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTOUT  DD   DISP=(NEW,PASS),UNIT=SYSDA,DCB=*.SORTIN,
//         SPACE=(1024,(5000,5000),RLSE),DSN=&&SORTO
//SYSOUT   DD   SYSOUT=*
//SYSIN    DD   *
 SORT FIELDS=(5,9,CH,A)
 INCLUDE COND=(5,1,CH,EQ,C'O',AND,14,1,CH,EQ,X'00')   BASE OWNERS ONLY
 SUM  FIELDS=NONE                                     ELIMINATE DUPS
//DEFINE   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYS1VOL  DD   DISP=SHR,UNIT=3380,VOL=SER=VSMN01
//SYS2VOL  DD   DISP=SHR,UNIT=3380,VOL=SER=VSMN02
//SYSIN    DD   *
  DEL DFRMM.NEW.SYS1
  DEL DFRMM.NEW.SYS2
  SET MAXCC=0
  DEF CL(NAME(DFRMM.NEW.SYS1) FILE(SYS1VOL)               -
         VOLUME(VSMN01) KILOBYTES(50000 9000) FSPC(20 20) -
         RECORDSIZE(512 9216)  KEYS(56 0) SHR(3 3) RUS)   -
      DATA(NAME(DFRMM.NEW.SYS1.DATA)                      -
           CISZ(10240) )                                  -
      INDEX(NAME(DFRMM.NEW.SYS1.INDEX) IMBED REPL         -
           CISZ(2048) )
  DEF CL(NAME(DFRMM.NEW.SYS2) FILE(SYS2VOL)               -
         VOLUME(VSMN02) KILOBYTES(50000 9000) FSPC(20 20) -
         RECORDSIZE(512 9216)  KEYS(56 0) SHR(3 3) RUS)   -
      DATA(NAME(DFRMM.NEW.SYS2.DATA)                      -
           CISZ(10240) )                                  -
      INDEX(NAME(DFRMM.NEW.SYS2.INDEX) IMBED REPL         -
           CISZ(2048) )
//*****************************************************************
//**              SOURCE SYSTEM                              **
//** COPY THE RECORDS FROM SOURCE TO BOTH TARGET CDSs         **
//*****************************************************************
//COPY     EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//OUTS1    DD   DSN=DFRMM.NEW.SYS1,DISP=SHR
//OUTS2    DD   DSN=DFRMM.NEW.SYS2,DISP=SHR
//SOURCE   DD   DSN=DFRMM.MASTER.CDS,DISP=SHR
//OWNER    DD   DSN=&&SORTO,DISP=(OLD,DELETE)
//SYSIN    DD   *
 REPRO OUTFILE(OUTS1) INFILE(SOURCE) FROMKEY(C) TOKEY(C) COUNT(1)
 REPRO OUTFILE(OUTS2) INFILE(SOURCE) FROMKEY(C) TOKEY(C) COUNT(1)
 REPRO OUTFILE(OUTS1) INFILE(SOURCE) FROMKEY(D) TOKEY(D)
 REPRO OUTFILE(OUTS2) INFILE(SOURCE) FROMKEY(D) TOKEY(D)
 REPRO OUTFILE(OUTS1) INFILE(SOURCE) FROMKEY(K) TOKEY(K)
 REPRO OUTFILE(OUTS2) INFILE(SOURCE) FROMKEY(K) TOKEY(K)
 REPRO INFILE(OWNER) OUTFILE(OUTS1)
 REPRO INFILE(OWNER) OUTFILE(OUTS2)
//*****************************************************************
//**              SOURCE TO A SINGLE TARGET                  **
```

```
//** COPY "P" RECORDS FROM THE SOURCE CDS TO THE ONE TARGET     **
//*****************************************************************
//COPYP2   EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//OUTS2    DD   DSN=DFRMM.NEW.SYS2,DISP=SHR
//SOURCE   DD   DSN=DFRMM.MASTER.CDS,DISP=SHR
//SYSIN    DD   *
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) FROMKEY(P) TOKEY(P)
//*****************************************************************
//**              SOURCE TO TWO TARGETS                          **
//** COPY "V" RECORDS BY VOLSER RANGE 100000-199999  TO SYS1     **
//** COPY "V" RECORDS BY VOLSER RANGE A00000-A00999  TO SYS2     **
//** COPY RACK RECORDS FOR THIS RANGE OF VOLUMES AS WELL         **
//*****************************************************************
//COPYV    EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//OUTS1    DD   DSN=DFRMM.NEW.SYS1,DISP=SHR
//OUTS2    DD   DSN=DFRMM.NEW.SYS2,DISP=SHR
//SOURCE   DD   DSN=DFRMM.MASTER.CDS
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'E500F1F0') COUNT(100000)
 /* copy volumes from V 100000       to  V  199999        */
 REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'C500F3F4F8F040404040F1F0') -
         TOKEY(X'C500F3F4F8F040404040F1F9F9F9F9F9')
 /*  COPY racks from E 3480    100000       to  E  3480    199999 */
 REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'C600F3F4F8F040404040F1F0') -
         TOKEY(X'C600F3F4F8F040404040F1F9F9F9F9F9')
 /*  COPY racks from F 3480    100000       to  F  3480    199999 */
 REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'E400F3F4F8F040404040F1F0') -
         TOKEY(X'E400F3F4F8F040404040F1F9F9F9F9F9')
 /*  COPY racks from U 3480    100000       to  U  3480    199999 */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'V 00C1F0') COUNT(1000)
   /*  COPY from V   A00000     to  V   A00999        */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'C500C3C1D9E3E2404040C1F0') -
         TOKEY(X'C500C3C1D9E3E2404040C1F0F0F9F9F9')
 /*  COPY racks from E CARTS   A00000       to  E  CARTS   A00999 */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'C600C3C1D9E3E2404040C1F0') -
         TOKEY(X'C600C3C1D9E3E2404040C1F0F0F9F9F9')
 /*  COPY racks from F CARTS   A00000       to  F  CARTS   A00999 */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'E400C3C1D9E3E2404040C1F0') -
         TOKEY(X'E400C3C1D9E3E2404040C1F0F0F9F9F9')
 /*  COPY racks from U CARTS   A00000       to  U  CARTS   A00999 */
//*****************************************************************
//**              SOURCE TO TWO TARGETS                          **
//** COPY BIN RECORDS FOR STORE VLT1  TO SYS1                    **
//** COPY BIN RECORDS FOR STORE VLT2  TO SYS2                    **
//*****************************************************************
//COPYB    EXEC PGM=IDCAMS,REGION=5000K
//SYSPRINT DD   SYSOUT=*
//OUTS1    DD   DSN=DFRMM.NEW.SYS1,DISP=SHR
//OUTS2    DD   DSN=DFRMM.NEW.SYS2,DISP=SHR
//SOURCE   DD   DSN=DFRMM.MASTER.CDS,DISP=SHR
//SYSIN    DD   *
```

```
REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'D9E4E5D3E3F140404040C3C1D9E3E2404040F0F0F0F0F0F0') -
         TOKEY(X'D9E4E5D3E3F140404040C3C1D9E3E2404040F0F0F0F9F9F9')
 /*  COPY bins from RUVLT1    CARTS   000000 for 1000 bins */
 REPRO INFILE(SOURCE) OUTFILE(OUTS1) -
       FROMKEY(X'E2E4E5D3E3F140404040C3C1D9E3E2404040F0F0F0F0F0F0') -
         TOKEY(X'E2E4E5D3E3F140404040C3C1D9E3E2404040F0F0F0F9F9F9')
 /*  COPY bins from RUVLT1    CARTS   000000 for 1000 bins */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'D9E4E5D3E3F240404040F3F4F8F040404040F0F0F0F0F0F0') -
         TOKEY(X'D9E4E5D3E3F240404040F3F4F8F040404040F0F0F1F9F9F9')
 /*  COPY bins from RUVLT1    3480    000000 for 2000 bins */
 REPRO INFILE(SOURCE) OUTFILE(OUTS2) -
       FROMKEY(X'E2E4E5D3E3F240404040F3F4F8F040404040F0F0F0F0F0F0') -
         TOKEY(X'E2E4E5D3E3F240404040F3F4F8F040404040F0F0F1F9F9F9')
 /*  COPY bins from RUVLT1    3480    000000 for 2000 bins */
//*****************************************************************
//**              EACH TARGET CDS                                **
//** RUN EDGUTIL WITH   PARM='MEND' TO UPDATE THE NEW TARGET    **
//**   CDS TO HAVE THE CORRECT COUNTS FOR RACK/BINS.             **
//**   ALSO SETS OWNER VOLUME COUNTS AND BUILDS OWNER VOLUME     **
//**   INFORMATION, AND DELETES UNUSED DATA SET RECORDS          **
//*****************************************************************
//MEND1    EXEC PGM=EDGUTIL,PARM='MEND'
//MASTER   DD   DISP=SHR,DSN=DFRMM.NEW.SYS1
//SYSPRINT DD   SYSOUT=*
//MEND2    EXEC PGM=EDGUTIL,PARM='MEND'
//MASTER   DD   DISP=SHR,DSN=DFRMM.NEW.SYS2
//SYSPRINT DD   SYSOUT=*
//*****************************************************************
//**              EACH TARGET CDS                                **
//** RUN EDGUTIL WITH   PARM='VERIFY(ALL)' ON THE NEW TARGET    **
//**   CDS TO VERIFY THE INTEGRITY OF THE NEW CDS.               **
//*****************************************************************
//VERIFY1  EXEC PGM=EDGUTIL,PARM='VERIFY(ALL)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   SUBSYS=(BLSR,'DDNAME=MASTERB')
//MASTERB  DD   DISP=SHR,DSN=DFRMM.NEW.SYS1
//* Use of Batch LSR can speed the running of EDGUTIL
//* IF you have not got BLSR setup on your system you must replac
//* the MASTER DD statement with the following one, and delete
//* the MASTERB DD statement
//*MASTER  DD   DISP=OLD,DSN=DFRMM.NEW.SYS1,
//*        AMP=('BUFND=50,BUFNI=50')
//SYSIN    DD   DUMMY
//VERIFY2  EXEC PGM=EDGUTIL,PARM='VERIFY(ALL)'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   SUBSYS=(BLSR,'DDNAME=MASTERB')
//MASTERB  DD   DISP=SHR,DSN=DFRMM.NEW.SYS2
//* Use of Batch LSR can speed the running of EDGUTIL
//* IF you have not got BLSR setup on your system you must replace
//* the MASTER DD statement with the following one, and delete
//* the MASTERB DD statement
//*MASTER  DD   DISP=OLD,DSN=DFRMM.NEW.SYS2,
//*        AMP=('BUFND=50,BUFNI=50')
//SYSIN    DD   DUMMY
```

Example 15-8 shows how you can create one backup procedure that works on both systems.

Example 15-8 Sample JCL to back up the DFSMSrmm CDSs and JOURNAL

```
//BACKCDS  PROC ID=
//BACKUP   EXEC PGM=EDGBKUP,PARM='BACKUP'
//SYSPRINT DD   SYSOUT=*
//MASTER   DD   DSN=DFRMM.MASTER.SYS&ID.,DISP=SHR
//JORUNAL  DD   DSN=DFRMM.JRNL.SYS&ID.,DISP=SHR
//BACKUP   DD   DSN=DFRMM.MASTER.BACKUP&ID.(+1),DISP=(,CATLG,DELETE),
//         UNIT=SYSDA,SPACE=(9216,(1000,999),RLSE),
//         LRECL=9216,DSORG=PS,BLKSIZE=0,RECFM=VB
//JRNLBKUP DD   DSN=DFRMM.JRNL.BACKUP&ID.(+1),DISP=(,CATLG,DELETE),
//         UNIT=SYSDA,SPACE=(9248,(1000,999),RLSE),
//         LRECL=9248,DSORG=PS,BLKSIZE=0,RECFM=VB
//         PEND
//BACKS1   EXEC BACKCDS,ID=1
//BACKS2   EXEC BACKCDS,ID=2
```

15.5.1 Converting to an existing CDS

When you convert information to DFSMSrmm and want the target to be an existing CDS, you are really merging databases. We suggest that you follow the normal conversion process as described in *Converting to Removable Media Manager: A Practical Guide,* SG24-4998. Follow this document to the point where the control data set is built and verified, you have started DFSMSrmm, and you have executed any commands that are necessary to complete the conversion. At that point, you know that the conversion is good and that you have a valid and complete CDS.

You can now follow the process that is explained in 15.4, “Merging the CDS” on page 605, to merge two DFSMSrmm CDSs.

If you have identified duplicate records, you might be able to change the extract and conversion control statements to avoid those duplicates instead of taking the actions that are suggested in 15.1.3, “Considerations” on page 596.

15.6 Merging DFSMSrmm environments

This section describes the following aspects of moving a system using DFSMSrmm into a sysplex where all the existing systems share a single DFSMSrmm database:

- The necessity of merging DFSMSrmm environments, or the possibility of having more than one DFSMSrmm environment on a single sysplex
- A checklist of points to consider if you decide to merge the DFSMSrmms
- A methodology for performing the merge
- A list of tools and documentation that can assist with this exercise

While some concepts in this chapter might apply to tape management systems other than DFSMSrmm, this chapter has been written specifically for DFSMSrmm by people familiar with that product. The term that we use in this chapter and in the index to describe systems sharing a single tape management system database is *RMMplex*. This is opposed to TAPEplex, TMSplex, or similar terms.

15.6.1 One or more RMMplexes

It *is* possible to have more than one RMMplex in a single sysplex. By definition, the RMMplex is the set of systems that share a single DFSMSrmm CDS.

If your target environment is a BronzePlex, the incoming system might share tape drives with systems in the target sysplex, but it does not share any DASD other than the volumes that contain the CDSs. If it does not share catalogs and the DFSMSrmm database (at a minimum, and probably the RACF database and storage management subsystem (SMS) environment), the incoming system is not part of the same RMMplex as the target sysplex.

If your target environment is a GoldPlex, the incoming system shares the system volumes (and therefore, potentially, the volume that contains the DFSMSrmm database) with the other systems in the target sysplex. However, because all the user catalogs, RACF, and so on are *not* shared, you will probably not merge your RMMplexes.

Finally, if your target environment is a PlatinumPlex, the incoming system *will* share the user DASD (and RACF, the storage management subsystem (SMS), and so on) with the other systems in the target sysplex. In this case, you *will* normally merge the RMMplexes.

15.6.2 Considerations for merging RMMplexes

This section contains (in checklist format) a list of elements that you must consider if you decide to merge two or more RMMplexes. We say *RMMplexes* instead of *DFSMSrmm CDSs*, because there is more to consider than just merging the contents of the DFSMSrmm CDSs.

Due to the interrelationship between the items in Table 15-4 on page 615, we strongly advise that you merge the SMS environment, the hierarchical storage management (HSM) environment, the security environment (for example, RACF), the catalog environment, and the DASD environment at the same time that you merge the DFSMSrmm environments.

Table 15-4 Considerations for merging RMMplexes

Considerations	Note	Type	Done
Ensure that toleration service is installed if necessary.	1	P	
Identify and eliminate duplicate records in the DFSMSrmm CDSs.	2	P	
Review PARMLIB options.	3	P	
Review DFSMSrmm procedure.	4	P	
Review DFSMSrmm CDS and journal data set sizes, and attributes.	5	P	
Tape configuration database (TCDB) tape CDS.	6	P	
Review any housekeeping jobs.	7	P	
Security considerations.	8	P	
Review DFSMSrmm exits.	9	P	

The Type specified in Table 15-4 relates to the sysplex target environment. The letter B represents a BronzePlex, G represents a GoldPlex, and P represents a PlatinumPlex. The steps in the following sections correspond to the rows in Table 15-4.

Step 1: Ensure that the toleration service is installed if necessary

Before you can start a DFSMSrmm merge process, check the DFSMSrmm software levels. If the DFSMSrmm software levels are different, check for compatibility PTFs and confirm that they are installed and applied.

To obtain a list of toleration and compatibility APARs for DFSMSrmm, see the DFSMS program directory, the DFSMSrmm PSP bucket, and Information APAR II08878.

Step 2: Identify and eliminate duplicate records within the CDSs

Note: Numerous references in this section are made to duplicate records and how to handle them. However, the first step is to *identify* whether you have duplicate records. The easiest way is to run the jobs in Example 15-9 on page 629 and Example 15-10 on page 629. The description of the first step of the job, “Step S001ICET” on page 626, discusses how the merge job handles the various record types, and specifically, what it does with any duplicate records that it finds. Look in the data sets that contain the duplicate records to see which record types you need to address.

Figure 15-7 on page 616 summarizes the process of merging multiple RMM CDSs. In this example, each LPAR has its own CDS. In our sysplex merge, we (hopefully) only have two: one from the incoming system and one from the target sysplex.

When merging CDSs, it is important to understand the contents of the CDSs. For example, if you merge two CDSs, you cannot merge them if the same resource is defined in each CDS. You must analyze the contents of the CDSs to ensure that there are no duplicate resources (or at least, no conflicting duplicates).

If you find that many duplicate tape VOLSERs exist, consider whether you really want to merge your DFSMSrmm CDSs. You might have many duplicate volumes and still want to merge the systems. In this case, before you perform the merge, consider using one of the many TAPECOPY utility programs available to copy the duplicate volumes to volumes with unique VOLSERs.

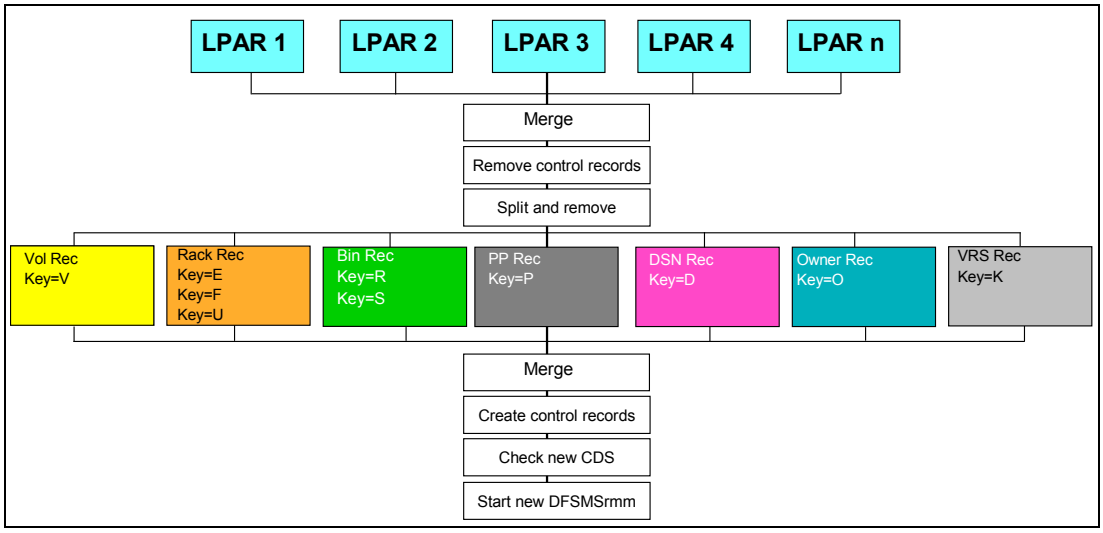


Figure 15-7 Merging process for multiple RMM CDSs

A sample merge job is shown in Example 15-10 on page 629. In addition to performing the merge, the job can also help to identify duplicate records in advance of the actual merge. The

job can be continually rerun until all duplicates (or at least those that require action) are resolved. The merge job is described in more detail in “Merging the CDSs” on page 626.

The merge job uses REPRO for merging the system’s DFSMSrmm CDSs into a flat file, sorts the file, discarding duplicate records in the process, and creates a new merged DFSMSrmm CDS. Duplicate records are written to SYSOUT files for you to investigate and take action. Note that all the duplicates that are deleted are from the same system. If you plan to use the merge process to remove duplicate records, all of the records that you want to discard *must* come from the same system. For example, you cannot use the merge to discard VRS records from RMMPLEX1 and volume records from RMMPLEX2.

15.6.3 CDS record types

This section describes each resource type in the CDS and explains what you can do to handle duplicate records.

The record types that are contained in the DFSMSrmm CDS are listed in Table 15-5.

Table 15-5 Record types in the DFSMSrmm CDS

Key type	Record type
C	Control
CA	Action
D	Data set
E	Empty rack
F	Scratch rack
K	VRS
O	Owner
P	Product
R	Empty bin
S	In-use bin
U	In-use rack
V	Volume

Volume record

The volume record contains detailed information about a volume and its contents. It also identifies the data sets that are on the volume and volume chaining information. Some fields in the records identify the existence of other records in the CDS.

When moving volume records from one CDS to another, you must be sure to move the related records, as well. These include the data set records, owner record, rack record, bin records, and product record. If you do not move them, the EDGUTIL utility might not be able to identify all the missing records, and therefore might not be able to create all the missing records.

When you merge CDSs, you must not have any duplicate volume serials. In fact, from an integrity point of view, you must not have duplicate volume serials in the same sysplex in any case, regardless of whether it causes a problem in DFSMSrmm. If you do, you must remove the duplicate volumes before starting the merge process. You need to understand why there are duplicate volumes, and, based on your understanding, select the best approach for dealing with them. While the merge job fixes some duplicate record problems by simply discarding one of the duplicates, you *must* ensure that there are no duplicate volume records before you start the merge. When you run the merge job, duplicate volume records are written to the file specified on the DUPSMAIN DD statement.

If you have duplicate volumes, there can be duplicate data set records for those volumes. If you delete the volumes, the related duplicate data set records are deleted automatically as part of the process of removing the duplicate volume.

Control record

Each DFSMSrmm CDS contains a control record. The control record contains the dates and times when the inventory management functions last ran, the counts of rack numbers and bin numbers for built-in storage locations, and the settings for some DFSMSrmm options.

When merging databases, only one control record is required. You must use the EDGUTIL program with PARM=MEND to correct the record counts after the merge job completes, and before you start DFSMSrmm using the new CDS.

The control record also contains the CDSID value. When merging the databases, retain the control record from the system that contained the CDSID that you want to continue to use (probably the one from DFSMSrmm on the target sysplex). All systems that use the DFSMSrmm CDSs must have a matching CDSID specified in the EDGRMMxx member of PARMLIB before being allowed to use the CDS.

As an alternative, you can bypass copying all control records. After you load the new DFSMSrmm CDS, you can create a new control record by using EDGUTIL with PARM=CREATE. After this is completed, use the same program with PARM=MEND to update the control record to correct the rack and bin counts.

Use the RMM LISTCONTROL CNTL subcommand to check the options that are in use, such as EXTENDEDBIN and STACKEDVOLUME. If the options that are in use differ between the CDSs, we suggest that you enable options to bring the CDSs in line with each other, and follow the steps to implement those options *before* starting the merge.

Action and move records

Action records contain information about the librarian actions that are outstanding. They include the release actions and any volume movements that are required. Duplicate records are deleted by the merge job, because there is a strong chance that they exist in each CDS, and the records are rebuilt by DFSMSrmm during inventory management after the merge.

Data set record

A data set record exists for each volume on which the related data set is written. Each record is uniquely identified by the VOLSER and physical file number of the data set. Remember that it is acceptable to have multiple tapes all containing the same data set (unlike DASD data sets where you generally try to avoid having the same data set name on more than one volume).

When merging CDSs, you want to ensure that there are no duplicate records across the two CDSs. Because the key of the data set record includes the VOLSER, avoiding duplicate VOLSERs ensures that there will be no duplicate data set records.

VRS record

The VRS records contain the retention and movement policies for your data. When merging the CDSs, you might find that you have the same policies defined in each CDS for some types of data. The policies are identified by fully-qualified or generic data set name and job name, VOLSER, and VOLSER prefix.

When you merge CDSs, check for and resolve any duplicate policies. To help you do this, use the RMM TSO SEARCHVRS subcommand to obtain a list of all policies. If you have a policy for the same data set name and job name combination in more than one CDS, check that the retention and movement defined are the same. If they are the same, you can use the policy

from any CDS. If they are not the same, you must resolve the conflict before you merge the CDSs. This way, duplicates can be ignored, regardless of which system the records are taken from.

The most important reason for checking the VRS definitions is to determine which VRS policy applies to a given data set and job name combination and what impact this can have after the merge. If you use generic VRS definitions, such as TOTHSM.**, it is possible that the VRS definition controlling a given data set and job can change even if you do not have duplicate policy names. This can potentially result in your keeping fewer generations of that data set or job than you intended. This is similar to the information in Chapter 13, “System Authorization Facility tape security” on page 475 and Appendix A, “Security topics” on page 633 about data sets being protected by different RACF profiles after a merge. Check all the VRS definitions for potential conflicts between VRS definitions. If you find a conflict, check, and, if necessary, modify the attributes of the VRS to meet your requirements.

There is a way to check whether the VRS records in the merged CDS will provide the results you expect. You can run EDGHSKP with PARM='VRSEL,VERIFY' using the test CDS that is created by the merge job. Specifying this PARM results in a trial run on inventory management. The resulting ACTIVITY file contains a list of all changes that will be made. By using the EDGJACTP sample job and looking at the summary of changes in matching VRS, you can see if you will have this problem.

Owner record

There is one owner record for each identified user that owns a DFSMSrmm resource, such as a volume or VRS. There can also be owner records for users that own no resources. Perhaps, you have defined these so that the information can be used within your installation.

The owner record contains detailed information about the user and the count of volumes that are owned. For users who own volumes, DFSMSrmm uses multiple owner records to provide directed retrieval of volume and data set information during search commands. You only need to copy the “Base” Owner records. When there are multiple owner records, there is no need to copy the Owner records that contain only volume information. When the records are loaded into the new CDS, your merge job uses the EDGUTIL utility with the PARM=MEND option to set the correct counts of owned volumes and build the owner volume records for owned volumes. A sample is shown in Step S004MEND in the “DFSMSrmm Merge job” in Example 15-10 on page 629.

Duplicate owner records are not a problem if the duplicate owners are the same user. In this case, duplicates can be safely ignored, regardless of which system the records are taken from.

If the duplicate owners are different users, you must decide how to handle that situation. You might have to assign different user IDs or define a new owner ID and transfer the owned resources to the new owner ID. Your objective is to have no duplicate owners, or to have any duplicates be for the same user in each case.

To add a new owner, type the command that is shown in Figure 15-8 on the RMMplex on which the owner is being changed.

```
RMM ADDOWNER new_owner_id DEPARTMENT(MERGE)
```

Figure 15-8 RMM ADDOWNER subcommand

To delete the duplicate owner and transfer volume ownership to the new owner ID (after you add the new owner ID), use the command that is shown in Figure 15-9 on the same RMMplex as the ADDOWNER command.

```
RMM DELETEOWNER old_owner_id NEWOWNER(new_owner-id)
```

Figure 15-9 RMM DELETEOWNER subcommand

Product record

A product record identifies a single product and software level and the volumes associated with that product. When you merge CDSs, the existence of duplicate product records can be resolved by running EDGUTIL MEND after the merge.

As an alternative to relying on MEND to resolve any duplicates, you can resolve the duplicates before the merge by choosing one of the product names to change. First, list all the volumes that are associated with the product name, using a command similar to the one that is in Figure 15-10.

```
RMM LISTPRODUCT oldproduct
```

Figure 15-10 RMM LISTPRODUCT subcommand

Then, create a new product name using the ADDPRODUCT command similar to the command that is shown in Figure 15-11 on page 620.

```
RMM ADDPRODUCT newproduct LEVEL(V01R01M00) NAME(.....)     -
       DESCRIPTION(.....) OWNER(....)
```

Figure 15-11 RMM ADDPRODUCT subcommand

Next, change all the volumes that were associated with the old product name to the new product name by using the CHANGEVOLUME command as shown in Figure 15-12.

```
RMM CHANGEVOLUME vvvvvv NUMBER(newproduct) FEATCD(A001) LEVEL(V01R03M00)
```

Figure 15-12 RMM CHANGEVOLUME subcommand

Finally, delete the old product name by using the DELETEPRODUCT command as shown in Figure 15-13.

```
RMM DELETEPRODUCT oldproduct LEVEL(V01R01M00)
```

Figure 15-13 RMM DELETEPRODUCT subcommand

If you do not resolve the duplicates before the merge, there are duplicate product records during the merge. All the records in the second CDS that is specified on the EXTRACT DD statement are discarded during the merge. The discarded records are written to the file specified on the DUPSMAIN DD statement. Therefore, if you decide to leave some duplicate records in the CDS so that they can be deleted during the merge, ensure that all those records are in the same CDS. After the merge, run EDGUTIL MEND to clean up the associations between the product and volume records.

Rack number record

There is a single rack record for each shelf location you defined to DFSMSrmm. Rack records are optional when the rack number and the volume serial number have the same value. There might be one for each volume defined. There can be more if you also defined empty rack numbers for use when volumes are added to the CDS. The rack number represents the volume's external VOLSER.

The DFSMSrmm record for each rack number can begin with one of three keys: E for empty, F for full, and U for in-use. It is possible that you can have duplicate rack numbers in the two CDSs without the merge job spotting them as duplicates. For example, rack number 200000 in one DFSMSrmm CDS can be Empty (so the key is E200000), and rack number 200000 in the DFSMSrmm CDS can be Full (so the key is F200000). However, the merge job ignores the key when identifying duplicate records so duplicate rack numbers *will* be identified.

When you merge CDSs, you must not have any duplicate rack numbers. Any duplicates must be deleted before the merge. Note that if there are duplicate records, the "U" records will be selected ahead of the "F" records, and finally the "E" records. Duplicates will be discarded, regardless of in which CDS each record might have resided before the merge. Any duplicates found at the time of the merge will be written to the file specified on the DUPSEFU DD statement.

Each rack number must have only one record type: either E, F, or U. You cannot have two record types for an individual rack number. If the duplicate already contains a volume, you can reassign the volume to another rack number by using the RMM CHANGEVOLUME subcommand. There are two different ways to reassign a volume to a new rack number as shown in Figure 15-14 on page 621.

```
RMM CHANGEVOLUME PP1234 POOL(A*)
RMM CHANGEVOLUME PP1234 RACK(AP1234)
```

Figure 15-14 RMM CHANGEVOLUME subcommand

In the first command, DFSMSrmm picks an empty rack in pool A*. In the second command, you select the empty rack to be used. Remember that the rack number is the external label on the tape, so changing large numbers of rack numbers is a significant effort. It helps if you can select a new rack number that is similar to the old one. For example, we changed rack number PP1234 in Figure 15-14 to rack number AP1234. Before cleaning up any duplicate rack numbers, see the description in 15.1.12, "Volume record" on page 602 for an explanation of how to handle duplicate volumes.

If you currently have rack numbers for volumes where the rack number and the volume serial number have the same value, you can choose to stop using rack numbers before starting the merge. This reduces the numbers of records in the CDS. To change a volume to no longer use a rack number, you can use these commands as shown in Figure 15-15.

```
RMM CHANGEVOLUME 100200 NORACK
RMM DELETERACK 100200
```

Figure 15-15 RMM CHANGEVOLUME and DELETERACK subcommands

Bin number record

There is a single bin record for each storage shelf location that you define to DFSMSrmm.

Like the rack numbers we just discussed, each bin number record can potentially have one of two different keys: in this case, R (empty) and S (in-use). Because the same bin can be

represented by two different keys, the merge jobs might not correctly identify all duplicates. For example, bin number 000029 can be represented by record R000029 in one CDS, and by record S000029 in the other. However, as for the rack numbers, the sample merge job is set up to ignore the key when searching for duplicate bin records, so duplicates are identified. They are written to the file specified on the DUPSRS DD statement. Note that if there are duplicate records, the “S” records are selected and the “R” records are discarded, regardless of in which CDS each record might have resided before the merge.

When you merge CDSs, you must not have any duplicate bin numbers. Any duplicates must be addressed before you perform the merge. If the duplicate already contains a volume, you can reassign the volume to another bin number by using the RMM CHANGEVOLUME subcommand. You can reassign a volume to a new bin number by issuing the following command as shown in Figure 15-16.

```
RMM CHANGEVOLUME A00123 BIN(000029)
```

Figure 15-16 RMM CHANGEVOLUME subcommands to assign a bin number

You can also remove the volume from the storage location, free the bin number, and delete the unused bin number. See the commands in Figure 15-17 on page 622 for an example of how to delete duplicate bin number X00022 in location VAULT1 of media name CARTS.

```
RMM CHANGEVOLUME A00123 LOCATION(HOME) CMOVE
RMM DELETEBIN X00022 MEDIANAME(CARTS) LOCATION(VAULT1)
```

Figure 15-17 RMM CHANGEVOLUME subcommands to change the location

The next time that you run inventory management, the volume is reassigned to the storage location, and another bin number is assigned.

Note: When you need to reassign bin numbers, consider how offsite movements are managed in your installation. If you depend on the volume being in a specific slot in a storage location, ensure that any volumes that you change are physically moved in the storage location.

15.6.4 Step 3: Review the DFSMSrmm PARMLIB options

To allow for a simplified backout process, we suggest that you do not change the current DFSMSrmm PARMLIB member, but instead create a new merged version. On the merge day, you can simply rename the members to pick up the merged definitions.

Review the IEFSSNxx member, and confirm that the DFSMSrmm subsystem name and definitions are the same on all systems that will share the same DFSMSrmm CDS. Also, review the EDGRMMxx member for differences and consolidate the members. Be aware of any changes you make that require supporting changes to RACF. Also, investigate whether all systems can share the same EDGRMMxx PARMLIB member as opposed to having a separate member for each DFSMSrmm instance.

The following EDGRMMxx options are affected by merging the CDSs:

- **OPTIONS**

  The options identify the installation options for DFSMSrmm. When you are merging CDSs, consolidate the OPTIONS command operands from all CDSs that you are merging to

ensure that you are using the same OPTIONS command operands on all DFSMSrmm subsystems.

After you merge DFSMSrmm CDSs, the CDSID of at least one DFSMSrmm subsystem will probably change. You must update the CDSID parameter to match the new CDSID, or if you have created a new DFSMSrmm CDS control record and you have not specified a CDSID, DFSMSrmm creates the ID in the control record from the CDSID automatically.

If the DFSMSrmm CDSs are kept in sync with the catalogs on the system, and you have enabled this using the inventory management CATSYNCH parameter, review the CATSYSID values specified for each system. If all systems sharing the merged DFSMSrmm CDS will also have access to the same user catalogs for tape data sets (typical case), you can use CATSYSID(*). However, if the catalogs are not fully shared, you must produce a consolidated list of system IDs for use with the CATSYSID and set the correct values in EDGRMMxx for each system.

Before you use the merged CDS, we suggest that you mark the CDS as not in sync with the catalogs. Use the EDGUTIL program with UPDATE and the SYSIN statements with CATSYNCH(NO) to do this. After you start DFSMSrmm using the newly merged CDS, you can reenable catalog synchronization using inventory management as detailed in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

If you use the VRSMIN option to specify a minimum number of VRSs, remember to update the value if you have a different number as a result of the merge.

- **LOCDEF**

  These options identify your locations to DFSMSrmm. When you merge CDSs, consolidate the LOCDEF options from all CDSs that you merge to ensure that you have no duplicates, and that no locations are left out. If you have duplicates, ensure that they are identified as the same location type and with the same media names.

- **VLPOOL**

  These options identify the ranges of rack numbers and pools of shelf space you have in your library. If your library is not changing, we suggest that you do not change your current VLPOOL options.

  If you merge CDSs and your existing VLPOOL definitions are different for each existing CDS, merge the definitions together to cover the merged sets of volumes and rack numbers. Ensure that the operands that are specified on each VLPOOL are correct when the same pool prefix is defined in more than one source environment.

- **REJECT**

  These options identify the ranges of volumes to be rejected for use on the system. The reject specifies the rack number prefix, or if a volume has no library shelf location, DFSMSrmm checks the volume serial number. The range of volumes that you specify on the REJECT statement needs to match the VLPOOL definitions that you have defined in Removable Media Manager (RMM).

  For example, you might currently specify that the incoming system can only use volumes beginning with A*, and the target sysplex systems can only use volumes beginning with B*. In this case, you need to update the REJECT statement at the time of the merge to indicate that all those systems can use tapes starting with either A* *or* B*.

  Where you have a tape library shared by multiple systems, define the same VLPOOL definitions to all systems and use the REJECT definitions to prevent systems from using volumes that are for use on other systems. The actions that you take for the REJECT definitions need to match how you handle the VLPOOL definitions.

  You can use the REJECT option for volumes defined to DFSMSrmm to control the use of volumes on the system. And for undefined volumes, you can use it to control the

partitioning of system-managed libraries. Consider both of these aspects as you merge the REJECT definitions.

- **OPENRULE**

  These options identify special processing for DFSMSrmm to perform during OPEN processing. It applies equally to both system-managed and non-system-managed volumes.

  In a shared CDS RMMplex, these commands allow you to control sharing volumes between systems while still maintaining a single CDS. OPENRULE allows you to specify what cannot be used as well as what can be used.

- **PRTITION**

  These options identify special processing for DFSMSrmm to perform during library entry, export/import, eject, CUA, OPEN, and EXPROC processing. The PRTITION commands apply to all volumes, not only system-managed volumes. However, only system-managed volumes benefit for the functions, such as library insert, CUA, and eject.

15.6.5 Step 4: Review the DFSMSrmm started task JCL

To allow for a simplified backout process, we suggest that you do not change the current DFSMSrmm procedure, but rather create a new merged version. On the merge day, you can rename the two procedures to perform the merge.

Review the DFSMSrmm procedure and investigate whether all systems can share the same DFSMSrmm procedure. This might be possible with little or no work because the only system-specific information that is contained in the DFSMSrmm procedure is the names of the PDO (EDGPDOx) data sets. PDO data sets are optional, but if you define them, they must be unique on each system. We suggest that you include the system name (&SYSNAME) in the data set name.

The DFSMSrmm CDS and journal names can be specified in either the DFSMSrmm JCL or in the PARMLIB member EDGRMMxx. For flexibility, we suggest that they are specified in EDGRMMxx, which allows you to easily change the CDS name by switching to a different PARMLIB member.

Also, check that the REGION size specified for the DFSMSrmm started procedure is large enough. Review the latest suggestions for this size in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Step 5: Review the CDS and journal data set sizes and attributes

To allow for a simplified backout process, we suggest that you create new versions of the DFSMSrmm CDS and journal data sets. On the merge day, you can rename the old ones, and rename the new ones to be ready to start DFSMSrmm after the merge.

Before you create the new data sets, calculate the new combined CDS size, and update the merge job with this information so that the CDS allocated in that job is valid. At the same time, calculate the new journal data set size, and create a job to allocate the data set.

Review the sample allocation jobs for the DFSMSrmm CDS, member EDGJMFAL in SYS1.SAMPLIB and for the DFSMSrmm journal, member EDGJNLAL in SYS1.SAMPLIB, to ensure that you are defining the new CDS with the latest information regarding record size, CISize, and Shareoptions. For more information about these data sets, see the *DFSMSrmm Implementation and Customization Guide*, SC26-7405.

Step 6: TCDB tape configuration data set

If the systems to be merged contain volume entries cataloged inside either a General Tape Volume Catalog (SYS1.VOLCAT.VGENERAL, for example) or a specific Tape Volume Catalog (SYS1.VOLCAT.Vx, for example), you need to list all the volume entries and create the volume entries in the target TCDB with the same attributes as in the current system. You can use a REPRO MERGECAT to copy entries from the old to the new volume catalogs, if required.

Confirm that the SMS automatic class selection (ACS) routines are allocating tape data sets to the appropriate storage groups associated with the tape libraries.

Review the DEVSUPxx member in SYS1.PARMLIB if the TCDB is being merged and ensure that all systems are entering the tapes with the same category number.

If the category number changes for any systems, there is no need to make any changes for private volumes. When the volume returns to scratch, its category changes to one of the new scratch categories. However, you must make changes for volumes already in scratch status. Look for more information in the “Processing Default Categories when using DEVSUPxx in an ATLDS” section in *DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries,* SC35-0427.

Obtain a list of volumes whose storage group name is *SCRTCH* using the ISMF Option 2.3 Mountable Tape. Use the ISMF ALTER command (not the line operator) to change the volume use attribute for all volumes in the list from scratch to scratch. This causes the library manager category for each volume to change to the new value established through DEVSUPxx.

Step 7: Review any housekeeping jobs

Review any housekeeping jobs and confirm that data set names are valid for the new merged environment. It might be possible to eliminate any DFSMSrmm jobs that were running on the incoming system previously, except for any report jobs that were not running on the target sysplex.

If you switched on the catalog synchronization feature, check that the CATSYSID definitions are correct and that the inventory management vital record processing is running on the correct system. Also, the inventory management expiration processing must be checked and running on multiple systems if not all user catalogs are shared (hopefully, in a PlatinumPlex, *all* catalogs are shared between all systems).

Step 8: Review security considerations

Chapter 8 in the *DFSMSrmm Implementation and Customization Guide*, SC26-7405, describes security for DFSMSrmm. If the systems to be merged are merging the RACF environment at the same time, and you have set up the PARMLIB definitions the same on all systems, there likely are no additional security concerns.

If either TAPEVOL or TAPEDSN is active, ensure that all EDGRMMxx definitions for TPRACF are the same. If you did not switch on TAPEVOL or TAPEDSN on all systems before you merged the DFSMSrmm CDSs, but now you will use them on all systems sharing the DFSMSrmm CDS, define all needed resources before the merged DFSMSrmm is started.

Step 9: Review DFSMSrmm exits

Determine what DFSMSrmm exits are in use (for example, EDGUX100, EDGUX200, CBRUXCUA, CBRUXENT, CBRUXEJC, and CBRUXVNL). Review the functions of these exits on the incoming system and the target sysplex systems. If they are not the same, see if

you can create a single version of each exit so that all systems will use identical exits, ensuring consistent processing.

15.6.6 Methodology for merging RMMplexes

You can choose from various approaches to merge RMMplexes. This section discusses one approach and provides some advice about the approach that provides the smallest risk or is the easiest to implement.

An alternate approach to the one discussed in this book is to create TSO RMM ADD... subcommands for all the volumes, data sets, racks, bins, and VRSs that are being merged. This approach can be used to avoid any outage to DFSMSrmm. Use care to ensure that the appropriate security options are already in place, for example, TAPEVOL/TAPEDSN(TVTOC). Since no tools are available to do this, you need to create these yourself.

The approach described in this document uses REPRO for the merging system's DFSMSrmm CDSs, sorts the resulting flat file, and creates a new merged DFSMSrmm CDS. Duplicate and unnecessary records are not included in the new CDS. Duplicates are written to sysout files for your review, and to confirm that you do not require them. DFSMSrmm is then restarted on all systems using the new merged DFSMSrmm CDS.

Merging the CDSs

Follow these steps to merge the CDSs:

1. Before the day of the merge, you need to have addressed all the issues and tasks identified in 15.6.2, "Considerations for merging RMMplexes" on page 615.
2. Stop DFSMSrmm on all systems in both RMMplexes. When you run the merge job, DFSMSrmm on all systems must be down. When you stop DFSMSrmm, do *not* use the OPT=RESET function because it allows tapes to be processed without DFSMSrmm's knowledge. You will not be able to use any tapes from this time until the merge is complete.
3. Back up both sets of CDSs and journals using the sample JCL in Example 15-9 on page 629. This JCL uses REPRO to back up the files. You must not change any of these files during the merge, but we take the backups to protect you from any accidental damage. Also, by using REPRO for the backup, we can use the backup files as input to the merge process.
4. If required, perform any merging of volume entries in the target Tape Catalog Database (TCDB).
5. Run the merge job that is shown in Example 15-10 on page 629 using the data sets that are created in Step 3.

   The sample merge job uses ICETOOL. If you do not have this program available, there is an alternate method using IDCAMS that is described in this chapter. If SYNCTOOL is available instead of ICETOOL, this job might run with little or no changes to the JCL and statements.

   The sample merge job can be run while RMM is running and while you are performing testing. However, all RMM subsystems need to be stopped before you run the merge job.

   The merge job consists of a number of steps. The steps, and the data sets used by each step, are explained:

   a. Step S001ICET

      The first step in the job simply deletes the output files that might be left from the last run of this job.

b. Step S001ICET

The next step in the job uses the ICETOOL utility of DFSORT. The input data sets are sequential unloads of the two DFSMSrmm CDSs (created with REPRO) and are specified on the DD EXTRACT statement. If duplicates are discovered in the EXTRACT files, the first occurrence is retained and the second and subsequent ones are discarded, with the exceptions previously noted for the rack and bin records. Therefore, the order of the DD statements on the EXTRACT is critical. The first DD needs to refer to the flat file from the RMMplex whose CDSID you want to retain. Whenever duplicate records are detected, the records from that RMMplex are retained and those from the other RMMplex are discarded. For this reason, ensure that any records that you want the merge job to discard are all in the same RMMplex.

The MAINCNTL DD statement contains the ICETOOL statements that will be used to sort and select most record types (C, K, P, and V). The associated SELECT statement in the TOOLIN DD specifies that duplicates must not be written to the output file for these record types (MAIN1S), but instead need to be written to DUPSMAIN for you to investigate and resolve.

The EFUXCNTL DD statement contains the ICETOOL statements that will be used to sort and select record types E, F, and U. They also ensure that the output file (EFU1S) will have only one record type for each rack. If the same rack number is discovered in more than one record type, records are selected with the “U” types first, followed by the “F” types, and finally the “E” types. Duplicate records are written to DUPSEFU for you to investigate and resolve.

The RSXXCNTL DD statement contains the ICETOOL statements that will be used to sort and select record types “R” and “S” into the data set specified on the RS1S DD statement. They also ensure that you have only one record type for each bin. If the same bin number is discovered in more than one record type, the “S” record is selected, and the “R” record is written to DUPSRS for you to investigate and resolve.

The OOOOCNTL DD statement contains the ICETOOL statements that will be used to sort and select type O records into the data set specified on the OO1S DD statement. The statement ensures that you have only one record for each owner. It also discards type O records that contain volume information (see “Owner record” on page 619).

Finally, the DDDDCNTL DD statement contains the ICETOOL statements that will be used to sort and select all the data set records. If you have addressed any duplicate volumes, there typically are no issues with copying all data set records. The key includes the VOLSER and the physical file number. Therefore, we have not provided a DUPSD DD statement as a destination for duplicate records.

c. Step S002ICET

This step also uses the ICETOOL utility of DFSORT. The input data sets are the temporary data set created by S001ICET that contains the (non-duplicate) CDS records. An output data set (DD MERGED) is created in the same format as the input data sets for S001ICET.

d. Step S003AMS

This step uses IDCAMS to define a new DFSMSrmm CDS, and REPROs the sequential file created in S002ICET (DD MERGED) into the new CDS. This is the CDS that is used in the newly merged RMMplex.

e. Step S004MEND

This step uses the EDGUTIL utility to repair all the record counts, set correct owners, and create missing records. Running EDGUTIL with the MEND parameter only updates the CDS referred to on the MASTER DD card, so you can safely run this utility

as many times as you like before the actual merge, *as long as you ensure that the MASTER DD points to your test CDS*.

If any of the volumes defined in the newly merged DFSMSrmm CDS are contained in an automated tape library (ATL) or Virtual Tape Server (VTS), the TCDB must already contain correct entries for these volumes. That is, SYS1.VOLCAT.VGENERAL must already have these volumes defined before the MEND can complete successfully. If all the volumes are inside an ATL or VTS, you can use the EDGUTIL MEND(SMSTAPE) function to add or correct the volumes in the TCDB.

> **Note:** If you specify SMSTAPE, the TCDB is updated, so do not use this option until you perform the actual merge.

f. Step S005MEND

If there are many records to be fixed, there can be a lot of output from EDGUTIL. To gain a clear understanding of which records the MEND was unable to fix (typically *very* few of these), run the EDGUTIL program again, this time with the VERIFY parameter. This identifies any actions that MEND was not able to take, and the ones for which you need to consider taking action. The message that you are most likely to see is `"VOLUME xxxxxx NOT FOUND IN VOLUME CATALOG"`. It is fixed when you run MEND with the SMSTAPE option.

6. Run EDGUTIL with UPDATE to turn off CATSYNCH. Turn off catalog synchronization before the merge is complete because the catalogs need to be resynchronized. Use the sample job in Example 15-11 on page 632.
7. Perform any renames of CDSs, Journals, PARMLIB members, started tasks, and JCL.
8. Start the DFSMSrmm procedure using the new PARMLIB member.
9. Check that the merged information is valid and correct by using DFSMSrmm commands, and the ISPF dialog to retrieve and display information.
10. Run EDGHSKP with CATSYNCH if catalog synchronization is to be maintained by DFSMSrmm. We suggest doing this for the performance benefits it can have for inventory management when policies are used on catalog retention. Do this before the first housekeeping job (EDGHSKP) is run after the merge.
11. If resynchronization between the newly merged DFSMSrmm CDS and TCDB is required, run the DFSMSrmm EDGUTIL program with a parameter of 'VERIFY(VOLCAT)'. In addition, if the systems in the RMMplex are all running OS/390 2.10 or later, specify 'VERIFY(SMSTAPE)'.
12. Run DFSMSrmm inventory management on the new CDS. During this run, DFSMSrmm VRSEL and DSTORE processing identifies volumes to be moved to the correct location. If any moves are identified, use EDGRPTD to produce the picking lists and actions of any moves that you know are still required.

    The inventory management run executes all inventory management functions, including backup.

Backout

If backout is required at any stage because you created new versions of the CDS, Journal, PARMLIB member and started task JCL, you can rename the old data sets back to the original names. There typically are no problems as a result of the additional Volume Entries you might have created in the TCDB. However, they need to be cleaned up before you attempt the merge process again.

15.6.7 Tools and documentation

This section provides information about tools and documentation that might help you complete this project.

Example 15-9 on page 629 contains sample JCL that you can use to back up the DFSMSrmm CDSs and journal data sets after the DFSMSrmm subsystems are stopped. This job does *not* reset the contents of the journal data sets. Even though we do not make any changes to any of these data sets during the merge process, it is prudent to make a backup of them before any changes are made, to protect yourself from any mistakes that might arise during the merge.

You can also use the JCL to create the input to the merge job for dummy runs to identify any duplicate records that might exist. However, if you run this JCL while the DFSMSrmm subsystems are running, you must comment out the //MASTER and //JOURNAL DD cards. If DFSMSrmm is *not* running when you run the EDGBKUP job, uncomment the MASTER and JOURNAL DD statements, and fill in the correct names for those data sets.

Example 15-9 JCL to back up the DFSMSrmm CDSs and journals

```
//S001BKUP EXEC PGM=EDGBKUP,PARM='BACKUP(NREORG)'
//SYSPRINT DD   SYSOUT=*
//*MASTER    DD  DISP=SHR,DSN=RMMPLEX1.RMM.CDS
//*JOURNAL   DD  DISP=SHR,DSN=RMMPLEX1.RMM.JOURNAL
//BACKUP    DD   DSN=hlq.RMMPLEX1.CDS.BACKUP,DISP=(,CATLG,DELETE),
//          DCB=(DSORG=PS,RECFM=VB,LRECL=9216),
//          UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//JRNLBKUP  DD   DSN=hlq.RMMPLEX1.JRNL.BACKUP,DISP=(,CATLG,DELETE),
//          DCB=(DSORG=PS,RECFM=VB,LRECL=32756,BLKSIZE=32760),
//          UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//S002BKUP  EXEC PGM=EDGBKUP,PARM='BACKUP(NREORG)'
//SYSPRINT  DD   SYSOUT=*
//*MASTER   DD   DISP=SHR,DSN=RMMPLEX2.RMM.CDS
//*JOURNAL  DD   DISP=SHR,DSN=RMMPLEX2.RMM.JOURNAL
//BACKUP    DD   DSN=hlq.RMMPLEX2.CDS.BACKUP,DISP=(,CATLG,DELETE),
//          DCB=(DSORG=PS,RECFM=VB,LRECL=9216),
//          UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//JRNLBKUP  DD   DSN=hlq.RMMPLEX2.JRNL.BACKUP,DISP=(,CATLG,DELETE),
//          DCB=(DSORG=PS,RECFM=VB,LRECL=32756,BLKSIZE=32760),
//          UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
```

Example 15-10 contains the merge job that we described earlier in this chapter. You can also use this job to identify duplicate records during the preparation for the merge.

Example 15-10 DFSMSrmm merge job

```
//CLEANUP EXEC PGM=IEFBR14
//DD1     DD   DSN=hlq.MERGED.RMM.MASTER.SEQ,DISP=(MOD,DELETE),
//        SPACE=(CYL,0),UNIT=SYSDA
//DD2     DD   DSN=hlq.MERGED.RMM.MASTER,DISP=(MOD,DELETE),
//        SPACE=(CYL,0),UNIT=SYSDA
/*
//S001ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD   SYSOUT=*
//DFSMSG   DD   SYSOUT=*
//DFSPARM  DD   *
VLSHRT
/*
//DUPSMAIN DD   SYSOUT=*
//DUPSEFU  DD   SYSOUT=*
```

```
//DUPSRS   DD   SYSOUT=*
//DUPSO    DD   SYSOUT=*
//EXTRACT  DD   DISP=SHR,DSN=hlq.RMMPLEX1.CDS.BACKUP
//         DD   DISP=SHR,DSN=hlq.RMMPLEX2.CDS.BACKUP
//MAIN     DD   DSN=&&MAIN,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//EFU      DD   DSN=&&EFU,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//RS       DD   DSN=&&RS,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//OOOO     DD   DSN=&&OOOO,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//DDDD     DD   DSN=&&DDDD,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//MAIN1S   DD   DSN=&&MAIN1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//EFU1S    DD   DSN=&&EFU1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//RS1S     DD   DSN=&&RS1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//OOO1S    DD   DSN=&&OOO1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//         UNIT=3390,DISP=(,PASS)
//TOOLIN   DD   *
  SORT FROM(EXTRACT) TO(MAIN) USING(MAIN)
  SELECT FROM(MAIN) TO(MAIN1S) ON(5,44,CH) FIRST -
   DISCARD(DUPSMAIN)
  SORT FROM(EXTRACT) TO(EFU) USING(EFUX)
  SELECT FROM(EFU) TO(EFU1S) ON(15,6,CH) FIRST -
   DISCARD(DUPSEFU)
  SORT FROM(EXTRACT) TO(RS) USING(RSXX)
  SELECT FROM(RS) TO(RS1S) ON(15,6,CH) FIRST -
   DISCARD(DUPSRS)
  SORT FROM(EXTRACT) TO(OOOO) USING(OOOO)
  SELECT FROM(OOOO) TO(OOO1S) ON(5,44,CH) FIRST -
   DISCARD(DUPSO)
  SORT FROM(EXTRACT) TO(DDDD) USING(DDDD)
//MAINCNTL DD   *
  OPTION EQUALS
  SORT FIELDS=(5,44,CH,A)
  INCLUDE COND=(5,2,CH,NE,C'CA',AND,5,2,CH,NE,C'CM',AND,
               5,1,CH,NE,C'O',AND,
               5,1,CH,NE,C'D',AND,
               5,1,CH,NE,C'R',AND,5,1,CH,NE,C'S',AND,
               5,1,CH,NE,C'E',AND,5,1,CH,NE,C'F',AND,5,1,CH,NE,C'U')
/*
//EFUXCNTL DD   *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,D,15,6,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'E',OR,5,1,CH,EQ,C'F',OR,5,1,CH,EQ,C'U')
/*
//RSXXCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,D,15,6,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'R',OR,5,1,CH,EQ,C'S')
/*
//OOOOCNTL DD   *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'O',AND,14,1,CH,EQ,X'00')
/*
```

```
//DDDDCNTL DD   *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'D')
/*
//S002ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD   SYSOUT=*
//DFSMSG   DD   SYSOUT=*
//DFSPARM  DD   *
VLSHRT
/*
//EXTRACT  DD   DISP=SHR,DSN=&&MAIN1S
//         DD   DISP=SHR,DSN=&&EFU1S
//         DD   DISP=SHR,DSN=&&RS1S
//         DD   DISP=SHR,DSN=&&OOO1S
//         DD   DISP=SHR,DSN=&&DDDD
//MERGED   DD   DSN=hlq.Merged.RMM.Master.Seq,
//         REFDD=*.EXTRACT,SPACE=(CYL,(xxx,yy),RLSE),
//         UNIT=SYSALLDA,DISP=(,CATLG)
//TOOLIN   DD  *
  SORT FROM(EXTRACT) TO(MERGED) USING(MOST)
//MOSTCNTL DD   *
  OPTION EQUALS
  SORT FIELDS=(5,56,CH,A)
/*
//S003AMS  EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT DD   SYSOUT=*,OUTLIM=1000000
//RMM      DD   DISP=SHR,DSN=hlq.Merged.RMM.Master.Seq
//SYSIN    DD   *
    DEFINE CLUSTER(hlq.Merged.RMM.Master)     -
          FILE(MASTER)                        -
          FREESPACE(15 0)                     -
          KEYS(56 0)                          -
          REUSE                               -
          RECSZ(512 9216)                     -
          SHR(3 3)                            -
          KILOBYTES(nnnn nnnn)                -
          STORAGECLASS(gspace)                -
          VOLUMES(volser))                    -
   DATA(NAME(hlq.Merged.RMM.Master.DATA)      -
          BUFFERSPACE(829440)                 -
          CISZ(26624))                        -
   INDEX(NAME(hlq.Merged.RMM.Master.INDEX)    -
          CISZ(2048))
   REPRO IFILE(RMM) ODS(hlq.Merged.RMM.Master)
/*
//*-------------------------------------------------------------------
//S004MEND  EXEC PGM=EDGUTIL,PARM='MEND'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN      DD   DUMMY
//*-------------------------------------------------------------------
//S005MEND  EXEC PGM=EDGUTIL,PARM='VERIFY'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN     DD   DUMMY
```

Example 15-11 contains sample JCL that you can use to turn off the CATSYNCH feature after you merge the CDSs, and before you start the DFSMSrmm subsystems using the new CDS.

Example 15-11 CATSYNCH job

```
//EDGUTIL   EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN     DD   *
  CONTROL CDSID(Merged) CATSYNCH(NO)
/*
```

The following documents contain helpful information as you proceed through this project:

- *DFSMSrmm Implementation and Customization Guide*, SC26-7405
- *DFSMSrmm Reporting*, SC26-7406
- *Merging Systems into a Sysplex,* SG24-6818

A

Security topics

This appendix provides the information that you need to implement your installation security policies in Data Facility System Managed Storage removable media manager (DFSMSrmm) using the FACILITY class profiles, and to activate RACF TAPEVOL and TAPEDSN classes. To protect DFSMSrmm functions, you need to use an external security product, such as RACF. You can use the information provided here, and the information provided in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405, and the *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683, for authorizing users and ensuring security.

While the focus is on RACF, other security products are covered where possible.

RACF implementation

This section takes you through the process of using RACF to implement security for DFSMSrmm resources and functions in your installation.

You must perform the following activities before you can implement DFSMSrmm:

- Assign a RACF user ID to DFSMSrmm
- Identify DFSMSrmm to RACF
- Define DFSMSrmm resources to RACF
- Define RACF groups for DFSMSrmm users
- Define DFSMSrmm resources to RACF
- Define the STGADMIN.ADR.DUMP.CNCURRNT resource

Note: You have to define the STGADMIN.ADR.DUMP.CNCURRNT resource only when you use DFSMSdss with concurrent copy.

Assigning a RACF user ID for DFSMSrmm

For a started procedure to access any of the system resources, a user ID must be associated with that task. The RACF user ID for DFSMSrmm does not need to match the name of the procedure name that you created in PROCLIB; any installation-selected RACF user ID is acceptable.

The user ID needs to have all the proper authorizations for accessing the system resources. We suggest defining a new user ID for each started procedure instead of using the default started task user ID (for example, STCUSER). In our system, we associated user ID `DFRMM` with the DFSMSrmm started procedure.

All data sets are created for use by the DFSMSrmm procedure. Add the DFSMSrmm RACF user ID to the access list for the data sets. We used the following RACF command shown in Figure A-1 to define the DFSMSrmm user ID.

```
ADDUSER DFRMM DFLTGRP(SYS1) NAME('DFSMSrmm Userid')
```

Figure A-1 Define a DFSMSrmm userid

Note: If you do not specify the DFLTGRP parameter on the ADDUSER command, the current connect group of the user issuing the ADDUSER command is used as the default.

Identifying DFSMSrmm to RACF

Before RACF 2.1, the only way to associate a started procedure with a RACF user ID was by coding the RACF started procedures table, ICHRIN03. With RACF 2.1, assigning RACF identities to started procedures has been greatly simplified by the introduction of the RACF STARTED class. You can add or modify security definitions for new and existing started procedures by issuing the RDEFINE and RALTER commands.

RACF STARTED class

The STARTED class allows you to assign RACF identities to started procedures dynamically using the RDEFINE and RALTER commands. Resource names in the STARTED class have the format *membername.jobname*. You assign identities, such as the RACF user ID and group ID, by using fields in the STDATA segment. You can define the started procedure resource, using either a generic profile name or a discrete profile name.

A *RACF generic profile* describes one or more data sets that have a similar name structure. A *RACF discrete profile* describes a specific data set on a specific volume. In our system, we created generic profiles for started procedures.

Issue the RACF commands shown in Figure A-2 to assign RACF identities to the DFSMSrmm started procedure and refresh the in-storage profiles, using the SETROPTS REFRESH command. The SETROPTS GENERIC command is needed only when you define generic profiles.

```
RDEFINE STARTED (DFRMM.*) UACC(NONE) STDATA(USER(DFRMM) GROUP(SYS1))
SETROPTS RACLIST(STARTED) REFRESH
SETROPTS GENERIC(STARTED) REFRESH
```

Figure A-2 Assign RACF identities to DFSMSrmm started procedure

Note: If you plan to use the EDGLABEL, EDGXPROC, or the EDGBKUP procedure, you must define the procedures in the RACF STARTED class.

Example A-1 shows a batch job that can be used to add the DFSMSrmm user ID in the RACF started class.

Example A-1 JCL to add DFSMSrmm user ID in RACF STARTED class

```
RACFRMM  JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
/* ************************************************************** */
/* CREATE AN ENTRY IN THE DYNAMIC STARTED PROCEDURE TABLE FOR DFRMM  */
/* ************************************************************** */
RDEFINE STARTED DFRMM.*           -
        STDATA(USER(DFRMM)        -
                 GROUP(SYS1)      -
                 PRIVELEGED(NO)   -
                 TRACE(NO)        -
                 TRUSTED(NO) )
```

ICHRIN03 started procedures table

We do not advise using the RACF started procedures table (ICHRIN03). The preferred way to add started procedure users to the RACF database is to use the STARTED class.

Defining DFSMSrmm resources to RACF

Define RACF FACILITY class profiles to control access to DFSMSrmm functions protected by the DFSMSrmm resource. The DFSMSrmm resources that you protect with RACF profiles in the FACILITY class each have an entity name prefixed with STGADMIN.EDG. Table A-1 lists the DFSMSrmm resources.

If you do not protect DFSMSrmm resources with RACF, an equivalent security product, or you do not have a security product in place, DFSMSrmm provides control for some resources as defined in the *DFSMSrmm Implementation and Customization Guide,* SC26-7405.

Table A-1 DFSMSrmm resources and protected functions

DFSMSrmm resource	Function protected
STGADMIN.EDG.ACTIONS.*action*[1]	Setting the release action.
STGADMIN.EDG.AV.*status.volser*[2]	Adding volumes.
STGADMIN.EDG.CD.COPYFROM.*dsname*[4]	Use of CHANGEDATASET COPYFROM subcommand to copy the data set attributes from one data set dsname to another data set.
STGADMIN.EDG.CD.VX[4]	Overriding DFSMSrmm VRSEL processing for a data set.
STGADMIN.EDG.CMOVE.*location.destination*	Confirmation of moves and ejects.
STGADMIN.EDG.CRLSE.action[1]	Confirmation of the release action.
STGADMIN.EDG.CV.[HOLD|NOHOLD].volser[3]	Setting and resetting the volume HOLD attribute.
STGADMIN.EDG.CV.RM[3]	Use of the RMM CHANGEVOLUME RETENTIONMETHOD subcommand to update the retention method for a volume. Use of the RMM CHANGEVOLUME RETAINBY subcommand to update the retain by attribute of a volume managed by the EXPDT retention method.
STGADMIN.EDG.DV.SCRATCH.*volser*	Deleting scratch volumes.
STGADMIN.EDG.FORCE	Changing information recorded by DFSMSrmm during Open/Close/End of Volume (O/C/EOV) processing. Use of the FORCE parameter that can be used in the ADDDATASET, CHANGEDATASET, CHANGEVOLUME, and DELETEDATASET subcommands to change information recorded by DFSMSrmm during O/C/EOV processing.
STGADMIN.EDG.EDGUPDT.UPDATE	Use of the EDGUPDT utility UPDATE function.
STGADMIN.EDG.HOUSEKEEP	Use of DFSMSrmm inventory management functions.
STGADMIN.EDG.HOUSEKEEP. REPEXT	Use to authorize the creation of report extract files when no other inventory management function is requested.

DFSMSrmm resource	Function protected
STGADMIN.EDG.IGNORE.TAPE. *volser*	Use of duplicate volume serial numbers or volume serial numbers not defined to DFSMSrmm and to allow a volume to be ignored. If you are authorized to ignore use of a tape volume, DFSMSrmm also overrides the Secure Authorization Facility (SAF) authorization for you to access data on the tape when the data is not defined to RACF and when the user is not authorized to the data. **Suggestion:** Do not assign an access level to the STGADMIN.EDG.IGNORE.TAPE.*volser* resource to any specific user group. Instead, wait until a tape volume that must be ignored by DFSMSrmm is identified, to define a resource granting a user or user group the needed access level. When the volume is no longer needed, delete the resource.
STGADMIN.EDG.IGNORE.TAPE.RMM.*volser*	Use of duplicate volume serial numbers and to allow a volume to be ignored. If you are authorized to ignore use of a tape volume, DFSMSrmm also overrides the SAF authorization for you to access data on the tape when the data is not defined to RACF and when the user is not authorized to the data. **Suggestion:** Do not assign an access level to the STGADMIN.EDG.IGNORE.TAPE.RMM.*volser* resource to any specific user group. Instead, wait until a tape volume that must be ignored by DFSMSrmm is identified, to define a resource granting a user or user group the needed access level. When the volume is no longer needed, delete the resource.
STGADMIN.EDG.IGNORE.TAPE. NORMM.*volser*	Use of volume serial numbers that are not defined to DFSMSrmm to allow a volume to be ignored. If you are authorized to ignore use of a tape volume, DFSMSrmm also overrides the SAF authorization for you to access data on the tape when the data is not defined to RACF and when the user is not authorized to the data. **Suggestion:** Do not assign an access level to the STGADMIN.EDG.IGNORE.TAPE.NORMM.*volser* resource to any specific user group. Instead, wait until a tape volume that must be ignored by DFSMSrmm is identified, to define a resource granting a user or user group the needed access level. When the volume is no longer needed, delete the resource.
STGADMIN.EDG.INIT	Setting of the INIT action.
STGADMIN.EDG.LABEL.*volser*	Creation of standard tape labels. The variable *volser* can be specified as a specific volume serial number or a generic volume serial number. For example, A12345 is a specific volume serial number and AB* is a generic volume serial number. If you use generic profiles, you can use these functions in a subset of your volumes. If the volume serial numbers and rack numbers match, you can control relabeling at the pool level, for example, you can have a pool using rack number prefix AB*. If you want to create an AL tape and your installation has an SL scratch pool, you need ALTER access to STGADMIN.EDG.LABEL.*volser*. The *volser* can be specified as the pool prefix of the scratch pool. If you want to switch to an AL tape from either an SL or NL tape that has already been assigned to you, UPDATE access to STGADMIN.EDG.LABEL.*volser* is required.

DFSMSrmm resource	Function protected
STGADMIN.EDG.LIST	List and search DFSMSrmm resources.
STGADMIN.EDG.LISTCONTROL	Use of the RMM LISTCONTROL subcommand to display DFSMSrmm control data set (CDS) control record information and EDGRMMxx PARMLIB settings.
STGADMIN.EDG.MASTER	Access to information in the DFSMSrmm CDS. Assign the CDS a universal access of NONE so that DFSMSrmm grants access to various functions through STGADMIN.EDG.MASTER.
STGADMIN.EDG.MOVES.*location.destination*	Initiation of moves and ejects.
STGADMIN.EDG.NOLABEL.*volser*	Creation of tapes without labels.
STGADMIN.EDG.OPERATOR	Use of the initialize, erase, and scan functions.
STGADMIN.EDG.OWNER. *owner*	Access to owned resources. DFSMSrmm checks this entity only if the command issuer is not the owner of the resource and does not have CONTROL access to STGADMIN.EDG.MASTER. Use of the RMM CHANGEVOLUME subcommand to update information based on the owner. Using STGADMIN.EDG.OWNER.*userid*, individual owners can permit other users to access owned volumes. An *owner* can be a group or department as well as an individual. Define owner resources only for those owners who will allow their volumes to be managed by another user.
STGADMIN.EDG.RELEASE	Use of the RMM DELETEVOLUME RELEASE subcommand to process any release actions specified for a volume.
STGADMIN.EDG.RESET.SSI	Use of the RESET facility for removing DFSMSrmm from the system. You can use the facility without defining this resource when you have no security product installed.
STGADMIN.EDG.VRS	Use of the RMM LISTVRS or SEARCHVRS subcommands to obtain information about vital record specifications (VRSs). Use of the RMM ADDVRS and DELETEVRS subcommands to define or remove VRSs.
STGADMIN.EDG.INERS.WRONGLABEL	Processing for volume mounted with the wrong label using the EDGINERS initialization process.

Notes:
[1] Action can be either SCRATCH, RETURN, REPLACE, NOTIFY, ERASE, or INIT.
[2] Status can be either SCRATCH, USER, MASTER, or VOLCAT.
[3] If you use a generic profile, the minimum non-generic profile name checked for by DFSMSrmm is STGADMIN.EDG.CV.
[4] If you use a generic profile, the minimum non-generic profile name checked for by DFSMSrmm is STGADMIN.EDG.CD.

Defining RACF groups for DFSMSrmm users

To implement RACF security for DFSMSrmm resources, define a set of user groups to handle different kinds of resources at different access levels. We suggest these function groups:

General user These persons might want to manage volumes they own and request information about resources defined to DFSMSrmm.

Storage administrator
These persons are responsible for specifying the policies to retain and move data sets and volumes.

System programmer The system programmer is responsible for the DFSMSrmm installation and implementation, and provides support for DFSMSrmm in the computing center.

Librarian The librarian creates scratch pull lists, moves the tapes between the built-in locations (LOCAL, DISTANT, REMOTE, and SHELF), installation-defined storage locations, and system-managed libraries.

Inventory management functions
The inventory management function is responsible for the daily and weekly DFSMSrmm inventory management.

Operator The operator handles DFSMSrmm system requests, including initializing and erasing tapes.

To implement RACF security for DFSMSrmm resources, we define a RACF group for each of the user groups, connect the user IDs to the appropriate function group, and permit only the RACF groups to the resources in the FACILITY class and to the DFSMSrmm data set profiles.

Table A-2 provides suggestions for authorizing different types of users. You need to implement security authorization to DFSMSrmm resources according to your working environments.

Table A-2 Access level for each group

Resource	General User	System Administrator	System Programmer	Librarian	Inventory Management Function	Operator
STGADMIN.EDG.ACTIONS.*action*	*	*	*	*	*	*
STGADMIN.EDG.AV.status.*volser*	*	*	*	*	*	*
STGADMIN.EDG.CD.COPYFROM.*dsname*	*	*	*	*	*	*
STGADMIN.EDG.CD.VX	*	*	*	*	*	*
STGADMIN.EDG.CMOVE.*location.destination*	*	*	*	*	*	*
STGADMIN.EDG.CRLSE.*action*	*	*	*	*	*	*
STGADMIN.EDG.CV.HOLD.*volser*	*	*	*	*	*	*
STGADMIN.EDG.CV.RM	*	*	*	*	*	*
STGADMIN.EDG.DV.SCRATCH.*volser*	*	*	*	*	*	*
STGADMIN.EDG.EDGUPDT.UPDATE	-					
STGADMIN.EDG.FORCE	-	U	-	U	-	-
STGADMIN.EDG.HOUSEKEEP	-	R	-	R	R	-
STGADMIN.EDG.HOUSEKEEP. REPEXT	-	-	-	-	-	-

Resource	General User	System Administrator	System Programmer	Librarian	Inventory Management Function	Operator
STGADMIN.EDG.IGNORE.TAPE. *volser*	tbd	tbd	tbd	tbd	tbd	tbd
STGADMIN.EDG.IGNORE.TAPE.RMM.*volser*	tbd	tbd	tbd	tbd	tbd	tbd
STGADMIN.EDG.IGNORE.TAPE. NORMM.*volser*	tbd	tbd	tbd	tbd	tbd	tbd
STGADMIN.EDG.INIT	*	*	*	*	*	*
STGADMIN.EDG.LABEL.*volser*	-	-	-	-	-	-
STGADMIN.EDG.LIST	*	*	*	*	*	*
STGADMIN.EDG.LISTCONTROL	*	*	*	*	*	*
STGADMIN.EDG.MASTER	R	C	R	C	R	R
STGADMIN.EDG.MOVES.*location.destination*	*	*	*	*	*	*
STGADMIN.EDG.CV.NOHOLD.*volser*	*	*	*	*	*	*
STGADMIN.EDG.NOLABEL.*volser*	-	-	-	-	-	-
STGADMIN.EDG.OPERATOR	-	U	R	U	-	U
STGADMIN.EDG.OWNER. *owner*	tbd	tbd	tbd	tbd	tbd	tbd
STGADMIN.EDG.RELEASE	R	R	R	R	R	R
STGADMIN.EDG.RESET.SSI	-	-	A	-	-	-
STGADMIN.EDG.VRS	R	C	R	R	R	R
STGADMIN.EDG.INERS.WRONGLABEL	-	C	-	C	-	-

Notes:
A Alter access
C Control access
R Read access
U Update access
- Not used
tbd To be defined
***** If this entity is not defined, access is based on STGADMIN.EDG.MASTER access.

Example A-2 shows a batch job to create the DFSMSrmm RACF groups and connect user IDs to the appropriate group. You can use this JCL as a base and modify it to fit your installation standards.

Example A-2 JCL to create RACF groups and connect user IDs

```
//RACFGRP  JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
```

```
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /*   DEFINE RACF ACCESS GROUP FOR ADMINISTRATOR                   */
 /* ************************************************************** */
   ADDGROUP EDGADMIN -
            SUBGROUP(SYS1) OWNER(SYS1)
      CONNECT Hallek   GROUP(EDGADMIN) AUTHORITY(USE)
      CONNECT WITTEN   GROUP(EDGADMIN) AUTHORITY(USE)
      CONNECT ROETTEN  GROUP(EDGADMIN) AUTHORITY(USE)
      CONNECT WELSCH   GROUP(EDGADMIN) AUTHORITY(USE)
      CONNECT JAEGER   GROUP(EDGADMIN) AUTHORITY(USE)
      CONNECT BERLING  GROUP(EDGADMIN) AUTHORITY(USE)
 /* ************************************************************** */
 /*   DEFINE RACF ACCESS GROUP FOR SYSTEM PROGRAMMER               */
 /* ************************************************************** */
   ADDGROUP EDGSYSPG -
            SUBGROUP(SYS1) OWNER(SYS1)
      CONNECT SCHLUM   GROUP(EDGSYSPG) AUTHORITY(USE)
      CONNECT SIEGEL   GROUP(EDGSYSPG) AUTHORITY(USE)
 /* ************************************************************** */
 /*   DEFINE RACF ACCESS GROUP FOR LIBRARIAN                       */
 /* ************************************************************** */
   ADDGROUP EDGLIB   -
            SUBGROUP(SYS1) OWNER(SYS1)
      CONNECT JOY      GROUP(EDGLIB) AUTHORITY(USE)
      CONNECT NADINE   GROUP(EDGLIB) AUTHORITY(USE)
      CONNECT ZERBINI  GROUP(EDGLIB) AUTHORITY(USE)
      CONNECT STRAUS   GROUP(EDGLIB) AUTHORITY(USE)
      CONNECT SEYFERT  GROUP(EDGLIB) AUTHORITY(USE)
 /* ************************************************************** */
 /*   DEFINE RACF ACCESS GROUP FOR INVENTORY MANAGEMENT FUNCTION   */
 /* ************************************************************** */
   ADDGROUP EDGJOB   -
            SUBGROUP(SYS1) OWNER(SYS1)
      CONNECT WILDEN   GROUP(EDGJOB) AUTHORITY(USE)
      CONNECT HENICKE  GROUP(EDGJOB) AUTHORITY(USE)
      CONNECT KEIL     GROUP(EDGJOB) AUTHORITY(USE)
 /* ************************************************************** */
 /*   DEFINE RACF ACCESS GROUP FOR INVENTORY MANAGEMENT FUNCTION   */
 /* ************************************************************** */
   ADDGROUP EDGOPER -
            SUBGROUP(SYS1) OWNER(SYS1)
      CONNECT TAUBER   GROUP(EDGOPER) AUTHORITY(USE)
      CONNECT ZINK     GROUP(EDGOPER) AUTHORITY(USE)
      CONNECT LOVELACE  GROUP(EDGOPER) AUTHORITY(USE)
```

RACF data set profiles

To protect the DFSMSrmm data sets, we define discrete profiles for the following data sets, and then permit the appropriate RACF groups to the following RACF data set profiles:

- DFSMSrmm control data set (CDS)
- DFSMSrmm journal data set
- DFSMSrmm message data set
- DFSMSrmm housekeeping data sets
- DFSMSrmm report data sets
- DFSMSrmm security report data sets

To protect all other DFSMSrmm data sets, define a generic data set profile using the high-level qualifier of your DFSMSrmm data sets.

Note: For enhanced generic naming considerations, see the *z/OS Security Server RACF Command Language Reference*, SA22-7687. The enhanced generic naming option applies only to data sets and allows you to use a double asterisk (**) in the DATASET class. It also changes the meaning of the single asterisk (*) at the end of a profile name.

Example A-3 shows a batch job to define all the DFSMSrmm data set profiles and permit the appropriate RACF groups and DFSMSrmm procedure to the data set profiles.

Example A-3 JCL to define DFSMSrmm RACF data set profiles

```
//RACFDSN  JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /*   DEFINE RACF GROUP                                            */
 /* ************************************************************** */
   ADDGROUP RMM  -
            SUBGROUP(SYS1) OWNER(SYS1)
 /* ************************************************************** */
 /*   DEFINE GLOBAL DATA SET PROFILE                               */
 /* ************************************************************** */
   ADDSD 'RMM.*' UACC(NONE)
      PERMIT 'RMM.*' -
             ID(EDGADMIN) ACC(NONE)
      PERMIT 'RMM.*' -
             ID(EDGLIB) ACC(NONE)
      PERMIT 'RMM.*' -
             ID(EDGJOB) ACC(NONE)
      PERMIT 'RMM.*' -
             ID(EDGOPER) ACC(NONE)
      PERMIT 'RMM.*' -
             ID(EDGSYSPG) ACC(NONE)
 /* ************************************************************** */
 /*   DEFINE CONTROL DATA SET DATA SET PROFILE                     */
```

```
/* ************************************************************** */
  ADDSD 'RMM.PROD.CDS' UACC(NONE) GENERIC
     PERMIT 'RMM.PROD.CDS' GENERIC-
            ID(EDGADMIN) ACC(READ)
     PERMIT 'RMM.PROD.CDS' GENERIC-
            ID(EDGLIB) ACC(READ)
     PERMIT 'RMM.PROD.CDS' GENERIC-
            ID(EDGSYSPG) ACC(ALTER)
/* ************************************************************** */
/*   DEFINE JOURNAL DATA SET DATA SET PROFILE                     */
/* ************************************************************** */
  ADDSD 'RMM.PROD.JRNL' UACC(NONE) GENERIC
     PERMIT 'RMM.PROD.JRNL' GENERIC-
            ID(EDGADMIN) ACC(READ)
     PERMIT 'RMM.PROD.JRNL' GENERIC-
            ID(EDGLIB) ACC(READ)
     PERMIT 'RMM.PROD.JRNL' GENERIC-
            ID(EDGSYSPG) ACC(ALTER)
/* ************************************************************** */
/*   DEFINE MESSAGE DATA SET DATA SET PROFILE                     */
/* ************************************************************** */
  ADDSD 'RMM.PROD.MSGS' UACC(NONE) GENERIC
     PERMIT 'RMM.PROD.MSGS' GENERIC-
            ID(EDGADMIN) ACC(UPDATE)
     PERMIT 'RMM.PROD.MSGS' GENERIC-
            ID(EDGLIB) ACC(UPDATE)
     PERMIT 'RMM.PROD.MSGS' GENERIC-
            ID(EDGSYSPG) ACC(ALTER)
/* ************************************************************** */
/*   DEFINE HOUSEKEEP DATA SET DATA SET PROFILE                   */
/* ************************************************************** */
  ADDSD 'RMM.HSKP.*' UACC(NONE)
     PERMIT 'RMM.HSKP.*' -
            ID(EDGADMIN) ACC(ALTER)
     PERMIT 'RMM.HSKP.*' -
            ID(EDGLIB) ACC(ALTER)
     PERMIT 'RMM.HSKP.*' -
            ID(EDGSYSPG) ACC(ALTER)
     PERMIT 'RMM.HSKP.*' -
            ID(EDGOPER) ACC(ALTER)
     PERMIT 'RMM.HSKP.*' -
            ID(EDGJOB) ACC(ALTER)
/* ************************************************************** */
/*   DEFINE REPORT DATA SET DATA SET PROFILE                      */
/* ************************************************************** */
  ADDSD 'RMM.REPORT.*' UACC(NONE)
     PERMIT 'RMM.REPORT.*' -
            ID(EDGADMIN) ACC(ALTER)
     PERMIT 'RMM.REPORT.*' -
            ID(EDGLIB) ACC(ALTER)
     PERMIT 'RMM.REPORT.*' -
            ID(EDGSYSPG) ACC(ALTER)
     PERMIT 'RMM.REPORT.*' -
            ID(EDGOPER) ACC(ALTER)
     PERMIT 'RMM.REPORT.*' -
```

```
            ID(EDGJOB) ACC(ALTER)
/* ************************************************************** */
/*   DEFINE SECURITY DATA SET DATA SET PROFILE                    */
/* ************************************************************** */
   ADDSD 'RMM.SECURITY.*' UACC(NONE)
      PERMIT 'RMM.SECURITY.*' -
            ID(EDGSYSPG) ACC(ALTER)
      PERMIT 'RMM.SECURITY.*' -
            ID(EDGJOB) ACC(ALTER)
      PERMIT 'RMM.SECURITY.*' -
            ID(AUDITOR) ACC(READ)
 /* ************************************************************** */
 /*   PERMIT DFRMM TO THE CONTROL DATA SET, JOURNAL DATA SET,      */
 /*    EXTRACT DATA SET, ACTIVITY LOG AND REPORT DATA SET          */
 /* ************************************************************** */
      PERMIT 'RMM.MASTER.CDS' GENERIC-
            ID(DFRMM) ACC(CONTROL)
      PERMIT 'RMM.MASTER.JOURNAL' GENERIC-
            ID(DFRMM) ACC(UPDATE)
      PERMIT 'RMM.MASTER.MSGS' GENERIC-
            ID(DFRMM) ACC(UPDATE)
      PERMIT 'RMM.HSKP*' -
            ID(DFRMM) ACC(UPDATE)
/*
```

RACF resource profiles

The following DFSMSrmm resources, described in Table A-1 on page 636, require RACF profiles in the FACILITY class:

- STGADMIN.EDG.ACTIONS.*action*
- STGADMIN.EDG.AV.*status.volser*
- STGADMIN.EDG.CD.COPYFROM.*dsname*
- STGADMIN.EDG.CD.VX
- STGADMIN.EDG.CMOVE.*location.destination*
- STGADMIN.EDG.CRLSE.*action*
- STGADMIN.EDG.CV.HOLD.*volser*
- STGADMIN.EDG.CV.RM
- STGADMIN.EDG.DV.SCRATCH.*volser*
- STGADMIN.EDG.EDGUPDT.UPDATE
- STGADMIN.EDG.FORCE
- STGADMIN.EDG.HOUSEKEEP
- STGADMIN.EDG.HOUSEKEEP. REPEXT
- STGADMIN.EDG.IGNORE.TAPE. *volser*
- STGADMIN.EDG.IGNORE.TAPE.RMM.*volser*
- STGADMIN.EDG.IGNORE.TAPE. NORMM.*volser*
- STGADMIN.EDG.INIT
- STGADMIN.EDG.LABEL.*volser*
- STGADMIN.EDG.LIST
- STGADMIN.EDG.LISTCONTROL
- STGADMIN.EDG.MASTER
- STGADMIN.EDG.MOVES.*location.destination*
- STGADMIN.EDG.CV.NOHOLD.*volser*
- STGADMIN.EDG.NOLABEL.*volser*
- STGADMIN.EDG.OPERATOR
- STGADMIN.EDG.OWNER. *owner*

- STGADMIN.EDG.RELEASE
- STGADMIN.EDG.RESET.SSI
- STGADMIN.EDG.VRS
- STGADMIN.EDG.INERS.WRONGLABEL

Example A-4 shows a batch job to define the resource profiles in the RACF FACILITY class and permits the RACF function groups as described in Table A-2 on page 639.

Example A-4 JCL to define DFSMSrmm resources in the FACILITY class

```
//RACFRDEF JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************* */
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************* */
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
/* ************************************************************* */
/* DEFINE RESOURCE STGADMIN.EDG.FORCE */
/* ************************************************************* */
 RDEFINE FACILITY STGADMIN.EDG.FORCE -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.FORCE -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.FORCE -
          ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.FORCE -
          ID(EDGLIB) ACC(READ) CLASS(FACILITY)
 /* ************************************************************* */
 /* DEFINE RESOURCE STGADMIN.EDG.HOUSEKEEP                        */
 /* ************************************************************* */
  RDEFINE FACILITY STGADMIN.EDG.HOUSEKEEP -
          UACC(NONE) OWNER(SYS1)
    RALT FACILITY STGADMIN.EDG.HOUSEKEEP -
         GLOBALAUDIT(ALL(UPDATE))
    PERMIT STGADMIN.EDG.HOUSEKEEP -
           ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
    PERMIT STGADMIN.EDG.HOUSEKEEP -
           ID(EDGLIB ) ACC(READ) CLASS(FACILITY)
    PERMIT STGADMIN.EDG.HOUSEKEEP -
           ID(EDGJOB ) ACC(READ) CLASS(FACILITY)
 /* ************************************************************* */
 /* DEFINE RESOURCE STGADMIN.EDG.HOUSEKEEP.RPTEXT                 */
 /* ************************************************************* */
  RDEFINE FACILITY STGADMIN.EDG.HOUSEKEEP.RPTEXT -
          UACC(NONE) OWNER(SYS1)
    RALT FACILITY STGADMIN.EDG.HOUSEKEEP.RPTEXT -
         GLOBALAUDIT(ALL(UPDATE))
    PERMIT STGADMIN.EDG.HOUSEKEEP.RPTEXT -
           ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
    PERMIT STGADMIN.EDG.HOUSEKEEP -
           ID(EDGLIB ) ACC(READ) CLASS(FACILITY)
```

```
   PERMIT STGADMIN.EDG.HOUSEKEEP.RPTEXT -
          ID(EDGJOB ) ACC(READ) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.*                   */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.* -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.IGNORE.TAPE.* -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.IGNORE.TAPE.* -
          ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.IGNORE.TAPE.* -
          ID(EDGJOB ) ACC(UPDATE) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.NORMM.*             */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
          ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.IGNORE.TAPE.NORMM.* -
          ID(EDGJOB ) ACC(UPDATE) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.RMM.*               */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.IGNORE.TAPE.RMM.* -
          ID(EDGSYSPG) ACC(READ) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.IGNORE.TAPE.RMM.* -
          ID(EDGJOB ) ACC(UPDATE) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.LABEL.*                         */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.LABEL.* -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.LABEL.* -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.LABEL.* -
          ID(EDGLIB) ACC(ALTER) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.LABEL.* -
          ID(EDGSYSPG) ACC(ALTER) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.LABEL.* -
          ID(EDGSYSPG) ACC(ALTER) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.LISTCONTROL                     */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.LISTCONTROL -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.LISTCONTROL -
        GLOBALAUDIT(ALL(UPDATE))
```

```
   PERMIT STGADMIN.EDG.LISTCONTROL -
          ID(EDGADMIN) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.LISTCONTROL -
          ID(EDGSYSPG) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.LISTCONTROL -
          ID(EDGLIB ) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.LISTCONTROL -
          ID(EDGJOB) ACC(CONTROL) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.MASTER                          */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.MASTER -
         UACC(READ) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.MASTER -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.MASTER -
          ID(EDGADMIN) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.MASTER -
          ID(EDGSYSPG) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.MASTER -
          ID(EDGLIB) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.MASTER -
          ID(EDGJOB) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.MASTER -
          ID(EDGOPER) ACC(CONTROL) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.NOLABEL.*                       */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.NOLABEL.* -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.NOLABEL.* -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.NOLABEL.* -
          ID(EDGADMIN) ACC(ALTER) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.NOLABEL.* -
          ID(EDGLIB ) ACC(ALTER) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.NOLABEL.* -
          ID(EDGOPER ) ACC(ALTER) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.OPERATOR                        */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.OPERATOR -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.OPERATOR -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.OPERATOR -
          ID(EDGSYSPG) ACC(UPDATE) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.OPERATOR -
          ID(EDGLIB) ACC(UPDATE) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.OPERATOR -
          ID(EDGOPER) ACC(UPDATE) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.OWNER.*                         */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.OWNER.* -
```

```
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.OWNER.* -
        GLOBALAUDIT(ALL(UPDATE))
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.RELEASE                         */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.RELEASE -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.RELEASE -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.RELEASE -
          ID(DFHSM) ACC(READ) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.RESET.SSI                       */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.RESET.SSI -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.RESET.SSI -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.RESET.SSI -
          ID(EDGSYSPG) ACC(ALTER) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.RESET.SSI -
          ID(EDGOPER) ACC(ALTER) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.VRS                             */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.VRS -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.VRS -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.VRS -
          ID(EDGADMIN) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.VRS -
          ID(EDGSYSPG) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.VRS -
          ID(EDGLIB) ACC(CONTROL) CLASS(FACILITY)
/* ************************************************************ */
/* DEFINE RESOURCE STGADMIN.EDG.INERS.WRONGLABEL                */
/* ************************************************************ */
 RDEFINE FACILITY STGADMIN.EDG.INERS.WRONG.LABEL -
         UACC(NONE) OWNER(SYS1)
   RALT FACILITY STGADMIN.EDG.INERS.WRONG.LABEL -
        GLOBALAUDIT(ALL(UPDATE))
   PERMIT STGADMIN.EDG.INERS.WRONG.LABEL -
          ID(EDGOPER) ACC(UPDATE) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.INERS.WRONG.LABEL -
          ID(EDGSYSPG) ACC(CONTROL) CLASS(FACILITY)
   PERMIT STGADMIN.EDG.INERS.WRONG.LABEL -
          ID(EDGLIB) ACC(CONTROL) CLASS(FACILITY)
```

Protecting DFSMSdss resources

You can use EDGHSKP or EDGBKUP with DFSMSdss DUMP, or the access method services (AMS) REPRO command to create a backup copy of your DFSMSrmm CDS and journal data set. To use DFSMSdss concurrent copy, you must define or update the STGADMIN.ADR.DUMP.CNCURRNT profile in the RACF FACILITY class.

Example A-5 shows a batch job to define the resource in the RACF FACILITY class.

Example A-5 JCL to define DFSMSdss RACF FACILITY class profile

```
//RACFRDEF JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.CNCURRNT                        */
 /* ************************************************************** */
   RDEFINE FACILITY STGADMIN.ADR.DUMP.CNCURRNT -
           UACC(NONE) OWNER(SYS1    )
      PERMIT STGADMIN.ADR.DUMP.CNCURRNT-
             ID(EDGADMIN) ACC(READ) CLASS(FACILITY)
```

Authorizing DFSMShsm to DFSMSrmm resources

DFSMSrmm can provide enhanced management functions for the tape volumes that DFSMShsm uses for each of its tape functions. DFSMSrmm treats DFSMShsm like any other tape user, not like a tape manager.

To run DFSMShsm with DFSMSrmm, DFSMShsm procedure names must be defined to RACF. Add the name of the DFSMShsm procedure to the RACF STARTED class or the started procedure table ICHRIN03. Use a DFSMShsm user ID other than the default user ID. For information about defining the DFSMShsm and aggregate backup and recovery support (ABARS) procedure names, see the *z/OS DFSMShsm Storage Administrators Guide*, SH35-0421.

Before you can use DFSMSrmm with DFSMShsm, you must authorize DFSMShsm to the following resources:

- STGADMIN.EDG.MASTER
- STGADMIN.EDG.RELEASE
- STGADMIN.EDG.OWNER.*hsmid*

If you have multiple DFSMShsm user IDs, for example, in a multisystem environment, and any DFSMShsm ID can return tapes to scratch status or to the DFSMShsm tape pool, you must authorize each DFSMShsm user ID. Define a resource profile STGADMIN.EDG.OWNER.*hsmid* for each DFSMShsm user ID, and give the other DFSMShsm user IDs UPDATE access to it.

When using DFSMSrmm with DFSMShsm, DFSMShsm needs access to the following resources to use scratch tapes:

- **STGADMIN.EDG.MASTER**: READ
- **STGADMIN.EDG.RELEASE**: READ
- **STGADMIN.EDG.OWNER.*hsmid*:** UPDATE

DFSMShsm needs access to the following resources to use a DFSMShsm-managed scratch tape pool:

- **STGADMIN.EDG.MASTER**: UPDATE
- **STGADMIN.EDG.OWNER.*hsmid*:** UPDATE

Example A-6 shows a batch job to add the name of the DFSMShsm procedure in the STARTED class, and permits the DFSMShsm user ID to the required DFSMSrmm RACF FACILITY class profiles.

Example A-6 JCL to define DFSMShsm to DFSMSrmm resources

```
//RACFHSM  JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the dynamic STARTED procedure table for DFHSM  */
 /* ************************************************************** */
   RDEFINE STARTED DFHSM.* -
           STDATA(USER(DFHSM ) -
                  GROUP(SYS1    ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
                  TRUSTED(NO) )
      /* ******************************************************** */
      /*   Permit DFHSM to STGADMIN.EDG.nnnnn    resources         */
      /* ******************************************************** */
      PERMIT STGADMIN.EDG.MASTER -
             ID(DFHSM) ACC(READ) CLASS(FACILITY)
      PERMIT STGADMIN.EDG.RELEASE -
             ID(DFHSM) ACC(READ) CLASS(FACILITY)
      PERMIT STGADMIN.EDG.OWNER.DFHSM*
             ID(DFHSM) ACC(UPDATE) CLASS(FACILITY)
```

Authorizing ABARS to DFSMSrmm resources

To run ABARS with DFSMSrmm, the ABARS procedure names must be defined to RACF. Add the name of the ABARS procedure to the RACF STARTED class or to the started procedure table ICHRIN03. For information about defining the DFSMShsm and ABARS procedure names to RACFR, see the *z/OS DFSMShsm Storage Administrators Guide*, SH35-0421.

To use DFSMSrmm with ABARS, you must authorize ABARS to the following resources:

- STGADMIN.EDG.MASTER
- STGADMIN.EDG.RELEASE
- STGADMIN.EDG.OWNER.*abarsid*

If you have multiple ABARS user IDs, for example, in a multisystem environment, and any ABARS ID can return tapes to scratch status, you must authorize each ABARS user ID. Define a RACF resource profile STGADMIN.EDG.OWNER.*abarsid* for each ABARS user ID, and give the other ABARS user IDs UPDATE access to it. This allows one ABARS to release the tapes initially obtained from scratch by another ABARS procedure.

The ABARS user ID needs access to the following resources to use DFSMSrmm with ABARS:

- **STGADMIN.EDG.MASTER**: READ
- **STGADMIN.EDG.RELEASE**: READ
- **STGADMIN.EDG.OWNER.*abarsid***: UPDATE

Example A-7 shows a batch job to add the name of the ABARS procedure in the STARTED class and permit the ABARS user ID to the required DFSMSrmm RACF FACILITY class resources.

Example A-7 JCL to define ABARS to DFSMSrmm resources

```
//RACFABAR JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the dynamic STARTED procedure table for ABARS  */
 /* ************************************************************** */
   RDEFINE STARTED ABARS.* -
           STDATA(USER(ABARS ) -
                  GROUP(SYS1   ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
                  TRUSTED(NO) )
      /* ******************************************************** */
      /*   Permit ABARS to STGADMIN.EDG.nnnnn    resources         */
      /* ******************************************************** */
      PERMIT STGADMIN.EDG.MASTER -
             ID(ABARS) ACC(READ) CLASS(FACILITY)
      PERMIT STGADMIN.EDG.RELEASE -
             ID(ABARS) ACC(READ) CLASS(FACILITY)
      PERMIT STGADMIN.EDG.OWNER.ADSM*
             ID(ABARS) ACC(UPDATE) CLASS(FACILITY)
      PERMIT STGADMIN.EDG.RESET
             ID(ABARS) ACC(READ) CLASS(FACILITY)
```

Authorizing EDGBKUP to DFSMSrmm resources

You use the OPTION JOURNALFULL(xx) command in PARMLIB member EDGRMMnn to define a percentage full threshold for the journal data set. When DFSMSrmm detects that the journal has reached this threshold, the procedure specified on the BACKUPPROC operand is started automatically. In this book, we use the name EDGBKUP for the backup procedure name.

The EDGBKUP procedure requires READ access to the resources shown in Figure A-3 to use the backup HOUSEKEEP function.

```
STGADMIN.EDG.HOUSEKEEP
STGADMIN.ADR.DUMP.CNCURRNT
```

Figure A-3 Resources to protect EDGBKUP function

This procedure also requires UPDATE and ALTER access to the data set profiles shown in Figure A-4.

```
RMM.HSKP.*          ALTER
RMM.PROD.MSG        UPDATE
```

Figure A-4 Data set profiles to protect EDGBKUP copies

Example A-8 shows a batch job to add the EDGBKUP procedure in the STARTED class, and permit it to the required RACF FACILITY class resources.

Example A-8 JCL to define EDGBKUP to DFSMSrmm resources

```
//RACFBKUP JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the STARTED procedure table for EDGBKUP       */
 /* ************************************************************** */
   RDEFINE STARTED EDGBKUP.* -
           STDATA(USER(DFRMM ) -
                  GROUP(SYS1    ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
                  TRUSTED(NO) )
      /* ************************************************************ */
      /*   Permit EDGHSKP to STGADMIN.EDG.nnnnn    resources          */
      /* ************************************************************ */
      PERMIT STGADMIN.EDG.HOUSEKEEP -
             ID(EDGBKUP) ACC(READ) CLASS(FACILITY)
      PERMIT 'RMM.PROD.MSGS' GENERIC -
             ID(EDGBKUP) ACC(UPDATE)
      PERMIT 'RMM.HSKP.*' -
```

```
            ID(EDGBKUP) ACC(ALTER)
      PERMIT STGADMIN.ADR.DUMP.CNCURRNT -
            ID(EDGBKUP) ACC(READ) CLASS(FACILITY)
```

Authorizing EDGXPROC to DFSMSrmm resources

Use the OPTION SCRATCHPROC command in PARMLIB member EDGRMMnn to specify the name of the procedure that you want to start automatically to replenish scratch volumes in an Automated Tape Library Dataserver (IBM 3494 or IBM 3495). In this book, we use the default procedure name of EDGXPROC by specifying SCRATCHPROC(EDGXPROC) in EDGRMMnn. A sample EDGXPROC procedure is shipped with DFSMSrmm.

When a manual or automatic tape library dataserver detects a low-on-scratch condition, where more scratch volumes are needed, object access method (OAM) issues a write-to-operator (WTO) message. DFSMSrmm intercepts the message and starts the procedure that you specify with the SCRATCHPROC value in PARMLIB.

The EDGXPROC procedure requires READ access to the resources as shown in Figure A-5 to use the expiration processing HOUSEKEEP function.

```
STGADMIN.EDG.HOUSEKEEP
```

Figure A-5 Resources to protect EDGHSKP function

This procedure also requires UPDATE access to the data set profile that is shown in Figure A-6.

```
RMM.PROD.MSG
```

Figure A-6 Data set profile to protect the message file

Example A-9 shows a batch job to add the EDGXPROC procedure in the STARTED class and permit it to the required RACF FACILITY class resources.

Example A-9 JCL to define EDGXPROC to DFSMSrmm resources

```
//RACFXPRC JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************* *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************* *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the dynamic STARTED procedure table for ABARS  */
 /* ************************************************************** */
   RDEFINE STARTED EDGXPROC.* -
           STDATA(USER(DFRMM ) -
                  GROUP(SYS1     ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
                  TRUSTED(NO) )
```

```
/* ********************************************************** */
/*   Permit EDGXPROC to STGADMIN.EDG.nnnnn    resources        */
/* ********************************************************** */
PERMIT STGADMIN.EDG.HOUSEKEEP -
       ID(EDGXPROC) ACC(READ) CLASS(FACILITY)
PERMIT 'RMM.PROD.MSGS' GENERIC-
       ID(EDGXPROC) ACC(ALTER)
```

Authorizing EDGLABEL to DFSMSrmm resources

You can run the EDGINERS utility as an operator-started procedure, so that the operator or librarian can make requests for tape labeling and erasing. The name of the procedure can be any valid alphanumeric procedure name from one to eight characters. This book uses EDGLABEL as the procedure name.

To automatically initialize or erase tapes, all volumes must be defined in the DFSMSrmm control data set, and the requested action must be set. In the automatic processing mode, EDGINERS requests that DFSMSrmm scan its control data set for the first 10 volumes waiting to be initialized or erased. If no volumes are found, the procedure ends without an error.

Note: DFSMSrmm erases volumes using the hardware security erase feature when supported.

The EDGLABEL procedure requires UPDATE access to the resource shown in Figure A-7 to use the expiration processing HOUSEKEEP function.

```
STGADMIN.EDG.OPERATOR
```

Figure A-7 Resource profile to protect the tape label function

Example A-10 shows a batch job to add the EDGLABEL procedure in the STARTED class and permit it to the required RACF FACILITY class resources.

Example A-10 JCL to define EDGLABEL to DFSMSrmm resources

```
//RACFLBL  JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the dynamic STARTED procedure table for LABEL  */
 /* ************************************************************** */
   RDEFINE STARTED EDGLABEL.* -
           STDATA(USER(DFRMM ) -
                  GROUP(SYS1     ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
```

```
                TRUSTED(NO) )
        /* ********************************************************** */
        /*   Permit EDGLABEL to STGADMIN.EDG.nnnnn    resources        */
        /* ********************************************************** */
        PERMIT STGADMIN.EDG.OPERATOR  -
               ID(EDGLABEL) ACC(READ) CLASS(FACILITY)
```

Authorizing EDGSCAN to DFSMSrmm resources

EDGINERS has been updated to support the reading and cross-verification of tape label information with the records defined in the DFSMSrmm CDS. The new SCAN function helps identify and manage tapes from other systems or that are in a problem state. The EDGINERS SCAN function reads the VOL1 and header labels for the first file on the specify volume.

When a manual or automatic tape library dataserver detects a low-on-scratch condition, where more scratch volumes are needed, OAM issues a WTO message. DFSMSrmm intercepts the message, and starts the procedure you specify with the SCRATCHPROC value in PARMLIB. The EDGSCAN procedure requires READ access to the resources as shown in Figure A-8 to read and cross-verify tape label information.

```
STGADMIN.EDG.OPERATOR
```

Figure A-8 Resources to protect EDGINERS function

Example A-11 shows a batch job to add the EDGXPROC procedure in the STARTED class and permit it to the required RACF FACILITY class resources.

Example A-11 JCL to define EDGSCAN to DFSMSrmm resources

```
//RACFSCAN JOB ,RMM,NOTIFY=SCHLUM,
//         MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),REGION=6M
//* ************************************************************** *
//*  Note
//*  ====
//*    To execute this job you need the RACF SPECIAL attribute
//* ************************************************************** *
//STEP01   EXEC PGM=IKJEFT01
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 /* ************************************************************** */
 /* Create an entry in the dynamic STARTED procedure table for ABARS  */
 /* ************************************************************** */
   RDEFINE STARTED EDGSCAN.* -
           STDATA(USER(DFRMM ) -
                  GROUP(SYS1     ) -
                  PRIVELEGED(NO) -
                  TRACE(NO) -
                  TRUSTED(NO) )
        /* ********************************************************** */
        /*   Permit EDGXPROC to STGADMIN.EDG.nnnnn    resources        */
        /* ********************************************************** */
        PERMIT STGADMIN.EDG.OPERATOR -
               ID(EDGSCAN) ACC(READ) CLASS(FACILITY)
```

CA-Top Secret implementation

The following sections include the necessary definitions for the CA-Top Secret product.

Defining user groups

Example A-12 shows the definitions of the different user groups for tape management.

Example A-12 CA-Top Secret user group definitions

```
 /* ************************************************************** */
 /*   DEFINE CA-Top Secret ACCESS GROUP FOR ADMINISTRATOR          */
 /* ************************************************************** */
TSS CRE(EDGADMIN) DEPT(XXXX) NAME('EDG ADMIN') TYPE(PROFILE)
TSS ADD(WOODY ) PROFILE(EDGADMIN)
TSS ADD(KUEHN ) PROFILE(EDGADMIN)
TSS ADD(GOGR  ) PROFILE(EDGADMIN)
TSS ADD(ETZ   ) PROFILE(EDGADMIN)
TSS ADD(BENGT ) PROFILE(EDGADMIN)
TSS ADD(NELSON) PROFILE(EDGADMIN)
 /* ************************************************************** */
 /*   DEFINE CA-Top Secret ACCESS GROUP FOR SYSTEM PROGRAMMER      */
 /* ************************************************************** */
TSS CRE(EDGSYSPG) DEPT(XXXX) NAME('EDG SYSPG') TYPE(PROFILE)
TSS ADD(SCHLUM  ) PROFILE(EDGSYSPG)
TSS ADD(SIEGEL  ) PROFILE(EDGSYSPG)
 /* ************************************************************** */
 /*   DEFINE CA-Top Secret ACCESS GROUP FOR LIBRARIAN              */
 /* ************************************************************** */
TSS CRE(EDGLIB) DEPT(XXXX) NAME('EDG LIBRARIAN') TYPE(PROFILE)
TSS ADD(JOY     ) PROFILE(EDGLIB)
TSS ADD(NADINE  ) PROFILE(EDGLIB)
TSS ADD(ZERBINI ) PROFILE(EDGLIB)
TSS ADD(STRAUS  ) PROFILE(EDGLIB)
TSS ADD(SEYFERT ) PROFILE(EDGLIB)
 /* ************************************************************** */
 /*   DEFINE CA-Top Secret GROUP FOR INVENTORY MANAGEMENT FUNCTION */
 /* ************************************************************** */
TSS CRE(EDGJOB) DEPT(XXXX) NAME('EDG JOB') TYPE(PROFILE)
TSS AD(HENICKE ) PROFILE(EDGJOB)
TSS AD(KEIL    ) PROFILE(EDGJOB)
TSS AD(BRIAN   ) PROFILE(EDGJOB)
 /* ************************************************************** */
 /*   DEFINE CA-Top Secret ACCESS GROUP FOR OPERATOR               */
 /* ************************************************************** */
TSS CRE(EDGOPER) DEPT(XXXX) NAME('EDG OPERATOR') TYPE(PROFILE)
TSS ADD(LOVELACE) PROFILE(EDGOPER)
TSS ADD(TAUBER  ) PROFILE(EDGOPER)
TSS ADD(ZINK    ) PROFILE(EDGOPER)
```

Protecting DFSMSrmm data sets

Example A-13 shows the profile definitions for the different DFSMSrmm data sets.

Example A-13 CA-Top Secret data set profile definitions

```
 /* ************************************************************** */
 /*   DEFINE GLOBAL DATA SET PROFILE                               */
 /* ************************************************************** */
TSS ADD(XXXX) DSN(RMM.)
TSS PE(EDGADMIN) DSN(RMM.*) ACCESS(NONE)
TSS PE(EDGLIB  ) DSN(RMM.*) ACCESS(NONE)
TSS PE(EDGJOB  ) DSN(RMM.*) ACCESS(NONE)
TSS PE(EDGOPER ) DSN(RMM.*) ACCESS(NONE)
TSS PE(EDGSYSPG) DSN(RMM.*) ACCESS(ALL)
 /* ************************************************************** */
 /*   DEFINE CONTROL DATA SET DATA SET PROFILE                     */
 /* ************************************************************** */
TSS PE(EDGADMIN) DSN(RMM.EGZB.CDS) ACCESS(UPDATE)
TSS PE(EDGLIB  ) DSN(RMM.EGZB.CDS) ACCESS(UPDATE)
TSS PE(EDGJOB  ) DSN(RMM.EGZB.CDS) ACCESS(UPDATE)
TSS PE(EDGOPER ) DSN(RMM.EGZB.CDS) ACCESS(UPDATE)
TSS PE(EDGSYSPG) DSN(RMM.EGZB.CDS) ACCESS(ALL)
 /* ************************************************************** */
 /*   DEFINE JOURNAL DATA SET DATA SET PROFILE                     */
 /* ************************************************************** */
TSS PE(EDGADMIN) DSN(RMM.EGZB.JRNL) ACCESS(UPDATE)
TSS PE(EDGLIB  ) DSN(RMM.EGZB.JRNL) ACCESS(UPDATE)
TSS PE(EDGJOB  ) DSN(RMM.EGZB.JRNL) ACCESS(UPDATE)
TSS PE(EDGOPER ) DSN(RMM.EGZB.JRNL) ACCESS(UPDATE)
TSS PE(EDGSYSPG) DSN(RMM.EGZB.JRNL) ACCESS(ALL)
 /* ************************************************************** */
 /*   DEFINE MESSAGE DATA SET DATA SET PROFILEDEFINE MESSAGE DATA S   */
 /* ************************************************************** */
TSS PE(EDGADMIN) DSN(RMM.HSKP.MESSAGE.DSET) ACCESS(UPDATE)
TSS PE(EDGLIB  ) DSN(RMM.HSKP.MESSAGE.DSET) ACCESS(UPDATE)
TSS PE(EDGJOB  ) DSN(RMM.HSKP.MESSAGE.DSET) ACCESS(UPDATE)
TSS PE(EDGOPER ) DSN(RMM.HSKP.MESSAGE.DSET) ACCESS(UPDATE)
TSS PE(EDGSYSPG) DSN(RMM.HSKP.MESSAGE.DSET) ACCESS(ALL)
 /* ************************************************************** */
 /*   DEFINE HOUSEKEEP DATA SET DATA SET PROFILE                   */
 /* ************************************************************** */
TSS PE(EDGSYSPG) DSN(RMM.HSKP.*) ACCESS(ALL)
TSS PE(EDGLIB  ) DSN(RMM.HSKP.*) ACCESS(ALL)
TSS PE(EDGJOB  ) DSN(RMM.HSKP.*) ACCESS(ALL)
TSS PE(EDGOPER ) DSN(RMM.HSKP.*) ACCESS(ALL)
 /* ************************************************************** */
 /*   DEFINE REPORT DATA SET DATA SET PROFILE                      */
 /* ************************************************************** */
TSS PE(EDGSYSPG) DSN(RMM.REPORT.*) ACCESS(ALL)
TSS PE(EDGSYSPG) DSN(RMM.REPORT.*) ACCESS(ALL)
TSS PE(EDGSYSPG) DSN(RMM.REPORT.*) ACCESS(ALL)
TSS PE(EDGSYSPG) DSN(RMM.REPORT.*) ACCESS(ALL)
```

Defining DFSMSrmm resources

Example A-14 shows the definitions for the DFSMSrmm resources.

Example A-14 CA-Top Secret DFSMSrmm resource definitions

```
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.HOUSEKEEP                      */
 /* ************************************************************** */
TSS ADD(XXXX) IBMFAC(STGADMIN)     IF STGADMIN not already defined
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.IGNORE.HOUSEKEEP) ACCESS(READ)
ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.IGNORE.HOUSEKEEP) ACCESS(READ)
ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.*                   */
 /* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.*) ACCESS(UPDATE)
ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.*) ACCESS(UPDATE)
ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.RMM.*                   */
 /* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.RMM.*) ACCESS(UPDATE)
ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.RMM*) ACCESS(UPDATE)
ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.IGNORE.TAPE.NORMM.*                   */
 /* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.NORMM.*) ACCESS(UPDATE)
ACTION(AUDIT)
TSS PE(EDGJOB ) IBMFAC(STGADMIN.EDG.IGNORE.TAPE.NORMM.*) ACCESS(UPDATE)
ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.LABEL.*                         */
 /* ************************************************************** */
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.LABEL.*) ACCESS(ALL) ACTION(AUDIT)
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.LABEL.*) ACCESS(ALL) ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.LABEL.*) ACCESS(ALL) ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.LISTCONTROL                     */
 /* ************************************************************** */
TSS PE(EDGADMIN) IBMFAC(STGADMIN.EDG.LISTCONTROL) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.LISTCONTROL) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.LISTCONTROL) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.LISTCONTROL) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.LISTCONTROL) ACCESS(CONTROL) ACTION(AUDIT)
  /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.MASTER                          */
 /* ************************************************************** */
TSS PE(EDGADMIN) IBMFAC(STGADMIN.EDG.MASTER) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.MASTER) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.MASTER) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.EDG.MASTER) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.MASTER) ACCESS(CONTROL) ACTION(AUDIT)
/* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.NOLABEL.*                       */
 /* ************************************************************** */
```

```
TSS PE(EDGADMIN) IBMFAC(STGADMIN.EDG.NOLABEL.*) ACCESS(ALL) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.NOLABEL.*) ACCESS(ALL) ACTION(AUDIT)
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.NOLABEL.*) ACCESS(ALL) ACTION(AUDIT)
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.OPERATOR                        */
/* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.OPERATOR) ACCESS(UPDATE) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.OPERATOR) ACCESS(UPDATE) ACTION(AUDIT)
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.OPERATOR) ACCESS(UPDATE) ACTION(AUDIT)
/* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.OWNER.*                         */
 /* ************************************************************** */
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.OWNER.*) ACCESS(UPDATE) ACTION(AUDIT)
TSS PE(EDGADMIN) IBMFAC(STGADMIN.EDG.OWNER.*) ACCESS(UPDATE) ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.RELEASE                         */
 /* ************************************************************** */
TSS PE(DFHSM   ) IBMFAC(STGADMIN.EDG.RELEASE) ACCESS(READ) ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.RESET.SSI                       */
 /* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.RESET.SSI) ACCESS(ALTER) ACTION(AUDIT)
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.RESET.SSI) ACCESS(ALTER) ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.VRS                             */
 /* ************************************************************** */
TSS PE(EDGADMIN) IBMFAC(STGADMIN.EDG.VRS) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.VRS) ACCESS(CONTROL) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.VRS) ACCESS(CONTROL) ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.INERS.WRONGLABEL                */
 /* ************************************************************** */
TSS PE(EDGOPER ) IBMFAC(STGADMIN.EDG.INERS.WRONGLABEL) ACCESS(UPDATE)
ACTION(AUDIT)
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.EDG.INERS.WRONGLABEL) ACCESS(CONTROL)
ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.EDG.INERS.WRONGLABEL) ACCESS(CONTROL)
ACTION(AUDIT)
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.ADR.DUMP.CNCURRNT                   */
 /* ************************************************************** */
TSS PE(EDGSYSPG) IBMFAC(STGADMIN.ADR.DUMP.CNCURRNT) ACCESS(READ) ACTION(AUDIT)
TSS PE(EDGLIB  ) IBMFAC(STGADMIN.ADR.DUMP.CNCURRNT) ACCESS(READ) ACTION(AUDIT)
TSS PE(EDGJOB  ) IBMFAC(STGADMIN.ADR.DUMP.CNCURRNT) ACCESS(READ) ACTION(AUDIT)
```

ACF2 implementation

The following sections include the necessary definitions for the ACF2 product.

Defining data set rules

Example A-15 shows the definition rules for the DFSMSrmm data sets.

Example A-15 ACF2 data set rule definitions

```
      ADDGROUP EDGADMIN  =  UIDSTRING1
      ADDGROUP EDGSYSPG  =  UIDSTRING2
      ADDGROUP EDGLIB    =  UIDSTRING3
      ADDGROUP EDGJOB    =  UIDSTRING4
      ADDGROUP EDGOPER   =  UIDSTRING5
/* ************************************************************** */
/*   DEFINE DATA SET RULE FOR HLQ 'RMM'                           */
/* ************************************************************** */
  $KEY(RMM)
   EGZB.CDS UID(UIDSTRING1) E(A) R(A) W(A)
   EGZB.CDS UID(UIDSTRING3) E(A) R(A) W(A)
   EGZB.CDS UID(UIDSTRING4) E(A) R(A) W(A)
   EGZB.CDS UID(UIDSTRING5) E(A) R(A) W(A)
   EGZB.CDS UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   EGZB.JRNL UID(UIDSTRING1) E(A) R(A) W(A)
   EGZB.JRNL UID(UIDSTRING3) E(A) R(A) W(A)
   EGZB.JRNL UID(UIDSTRING4) E(A) R(A) W(A)
   EGZB.JRNL UID(UIDSTRING5) E(A) R(A) W(A)
   EGZB.JRNL UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   HSKP.MESSAGE.DSET UID(UIDSTRING1) E(A) R(A) W(A)
   HSKP.MESSAGE.DSET UID(UIDSTRING3) E(A) R(A) W(A)
   HSKP.MESSAGE.DSET UID(UIDSTRING4) E(A) R(A) W(A)
   HSKP.MESSAGE.DSET UID(UIDSTRING5) E(A) R(A) W(A)
   HSKP.MESSAGE.DSET UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   HSKP.* UID(UIDSTRING3) E(A) R(A) W(A) A(A)
   HSKP.* UID(UIDSTRING4) E(A) R(A) W(A) A(A)
   HSKP.* UID(UIDSTRING5) E(A) R(A) W(A) A(A)
   HSKP.* UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   REPORT.* UID(UIDSTRING3) E(A) R(A) W(A) A(A)
   REPORT.* UID(UIDSTRING4) E(A) R(A) W(A) A(A)
   REPORT.* UID(UIDSTRING5) E(A) R(A) W(A) A(A)
   REPORT.* UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   * UID(*) UID(UIDSTRING2) E(A) R(A) W(A) A(A)
   * UID(*)
```

The following descriptions refer to Example A-15:

dsnmask
Specifies the name of the data set or a mask to which this entry applies. This parameter is required. For example, WORK.MASTER represents the name of the data set PAYROLL.WORK.MASTER. (The $KEY or the $PREFIX control statement specifies the high-level index PAYROLL.) You must specify the *dsnmask* parameter before any other parameters in a rule entry.

Read(Allow|Log|Prevent)
Specifies read access and the action A, L, or P that you want eTrust CA-ACF2 to take when the environment matches. Before this access permission applies, the actual access attempt must match the environment defined by other parameters of the access rule entry. The letter codes are defined this way:

A Allow the access
L Permit the access but log the event
P Prevent the access

If not specified in the rule entry, this access permission defaults to READ(P). ACF99900 messages are not issued for data set read accesses.

Write(Allow|Log|Prevent)
Specifies write access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter works similarly to how the READ parameter works. If not specified in the rule entry, this access permission defaults to WRITE(P).

Allocate(Allow|Log|Prevent)
Specifies allocate access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter specifies that a user has create, delete, rename, and catalog authority to a data set. If not specified in the rule entry, this access permission defaults to ALLOC(P).

Execute(Allow|Log|Prevent)
Specifies execute access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter works similarly to how the READ parameter does. However, its access permission is the specified value or the value of the READ parameter, whichever designates the most permissive access. (For example, if READ(P) and EXEC(L) are specified, EXEC(L) applies.) If READ access specifies A or L and EXEC access is not specified, it defaults to the same access as READ. Also, if READ access is less restrictive than EXEC access (for example, R(A) E(L)), EXEC access will be set to the same access as READ.

Defining DFSMSrmm resources

Example A-16 shows the definitions for the DFSMSrmm resources.

Example A-16 ACF2 DFSMSrmm resource definitions

```
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.HOUSEKEEP                      */
 /* ************************************************************** */
$KEY(STGADMIN.EDG.HOUSEKEEP) TYPE(FAC)
UID(UIDSTRING2) SERVICE(READ) ALLOW
UID(UIDSTRING4) SERVICE(READ) ALLOW
UID(UIDSTRING5) SERVICE(READ) ALLOW
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.IGNORE.TAPE.VOLSER.*
/ * ************************************************************** */
$KEY(STGADMIN.EDG.IGNORE.TAPE.VOLSER.*) TYPE(FAC)
UID(UIDSTRING2) SERVICE(READ) ALLOW
UID(UIDSTRING4) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.IGNORE.TAPE.RMM.VOLSER.*
/ * ************************************************************** */
$KEY(STGADMIN.EDG.IGNORE.TAPE.RMM.VOLSER.*) TYPE(FAC)
UID(UIDSTRING2) SERVICE(READ) ALLOW
UID(UIDSTRING4) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.IGNORE.TAPE.NORMM.VOLSER.*
/ * ************************************************************** */
```

```
$KEY(STGADMIN.EDG.IGNORE.TAPE.NORMM.VOLSER.*) TYPE(FAC)
UID(UIDSTRING2) SERVICE(READ) ALLOW
UID(UIDSTRING4) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.LABEL.*
/* ************************************************************** */
$KEY(STGADMIN.EDG.LABEL.*) TYPE(FAC)
UID(UIDSTRING1) LOG
UID(UIDSTRING2) LOG
UID(UIDSTRING5) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.LIST.CONTROL
/* ************************************************************** */
$KEY(STGADMIN.EDG.LISTCONTROL) TYPE(FAC)
UID(UIDSTRING1) SERVICE(UPDATE) LOG
UID(UIDSTRING2) SERVICE(UPDATE) LOG
UID(UIDSTRING3) SERVICE(UPDATE) LOG
UID(UIDSTRING4) SERVICE(UPDATE) LOG
UID(UIDSTRING5) SERVICE(UPDATE) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.EDG.MASTER
/* ************************************************************** */
$KEY(STGADMIN.EDG.MASTER) TYPE(FAC)
UID(UIDSTRING1) SERVICE(UPDATE) LOG
UID(UIDSTRING2) SERVICE(UPDATE) LOG
UID(UIDSTRING3) SERVICE(UPDATE) LOG
UID(UIDSTRING4) SERVICE(UPDATE) LOG
UID(UIDSTRING5) SERVICE(UPDATE) LOG
 /* ************************************************************** */
 /*   DEFINE RESOURCE STGADMIN.EDG.NOLABEL.*                       */
 /* ************************************************************** */
$KEY(STGADMIN.EDG.NOLABEL.*) TYPE(FAC)
UID(UIDSTRING1) LOG
UID(UIDSTRING3) LOG
UID(UIDSTRING5) LOG
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.MASTER                          */
/* ************************************************************** */
$KEY(STGADMIN.EDG.MASTER) TYPE(FAC)
UID(UIDSTRING2) SERVICE(READ,UPDATE) LOG
UID(UIDSTRING3) SERVICE(READ,UPDATE) LOG
UID(UIDSTRING5) SERVIVE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.OWNER.*                         */
/* ************************************************************** */
$KEY(STGADMIN.EDG.OWNER) TYPE(FAC)
UID(UIDSTRING1) SERVICE(READ,UPDATE) LOG
UID(UIDSTRING3) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.RELEASE                         */
/* ************************************************************** */
$KEY(STGADMIN.EDG.RELEASE) TYPE(FAC)
UID(DFHSM) SERVICE(READ) ALLOW
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.RESET.SSI                       */
```

```
/* ************************************************************** */
$KEY(STGADMIN.EDG.RESET.SSI) TYPE(FAC)
UID(UIDSTRING2) LOG
UID(UIDSTRING5) LOG
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.VRS                             */
/* ************************************************************** */
$KEY(STGADMIN.EDG.VRS) TYPE(FAC)
UID(UIDSTRING1) SERVICE(READ,UPDATE) LOG
UID(UIDSTRING2) SERVICE(READ,UPDATE) LOG
UID(UIDSTRING3) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE RESOURCE STGADMIN.EDG.INERS.WRONGLABEL                */
/* ************************************************************** */
$KEY(STGADMIN.EDG.INERS.WRONGLABEL) TYPE(FAC)
UID(UIDSTRING1) LOG
UID(UIDSTRING2) LOG
UID(UIDSTRING5) SERVICE(READ,UPDATE) LOG
/* ************************************************************** */
/*   DEFINE ACF2 RESOURCE STGADMIN.ADR.DUMP.CNCURRNT              */
/* ************************************************************** */
$KEY(STGADMIN.EDG.INERS.WRONGLABEL) TYPE(FAC)
UID(UIDSTRING1) SERVICE(READ) ALLOW
UID(UIDSTRING2) SERVICE(READ) ALLOW
UID(UIDSTRING4) SERVICE(READ) ALLOW
```

Glossary

A

Abend. Abnormal end of task.

ACS. Automatic class selection.

AL. American National Standards Label.

AMODE. Addressing mode.

AMS. Access method services.

ANDVRS. An RMM ADDVRS TSO subcommand operand. See Using AND.

ANSI. American National Standards Institute.

APAR. Authorized program analysis report.

APF. Authorized program facility.

API. Application Programming Interface.

ASA. American Standards Association.

Assigned date. The date that the volume is assigned to the current owner. Assigned date is not meaningful for a scratch volume.

ATL. Automated tape library.

AUL. ANSI and user header or trailed label.

Automated tape library. A device consisting of robotic components, cartridge storage areas, tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. See also tape library. Contrast with manual tape library.

Automatic cartridge loader. An optional feature of the 3480 Magnetic Tape Subsystem that allows reloading of multiple tape cartridges. This feature is standard in the 3490 Magnetic Tape Subsystem.

Automatic recording. In DFSMSrmm, the process of recording information about a volume and the data sets on the volume in the DFSMSrmm control data set at open or close time.

Availability. For a storage subsystem, the degree to which a data set or object can be accessed when requested by a user.

B

B. Bytes.

Backup. The process of creating a copy of a data set or object to be used in case of accidental loss.

Basic catalog structure. The name of the catalog structure in the integrated catalog facility environment. See also integrated catalog facility catalog.

BCS. Basic catalog structure.

Bin number. The specific shelf location where a volume resides in a storage location; equivalent to a rack number in the removable media library. See also shelf location.

BLP. Bypass label processing.

BTLS. Basic Tape Library Support.

Built-in storage location. One of the DFSMSrmm defined storage locations: LOCAL, DISTANT, and REMOTE.

BVIR. Bulk volume information retrieval.

C

Cache fast write. A storage control capability in which the data is written directly to cache without using non-volatile storage. Cache fast write is useful for temporary data that is readily recreated., such as the sort work files created by DFSORT. Contrast with DASD fast write.

Cartridge eject. For an IBM 3494 or IBM 3495 Tape Library Data set, the act of physically removing a tape cartridge usually under robot control, by placing it in an output station. The software logically removes the cartridge by deleting or updating the tape volume record in the tape configuration database. For a manual tape library dataserver, the act of logically removing a tape cartridge from the manual tape library dataserver by deleting or updating the tape volume record in the tape configuration database.

Cartridge entry. For either an IBM 3494 Tape Library Dataserver, IBM 3495 Tape Library Dataserver, or a IBM Model M10 3495 Tape Library Dataserver, the process of logically adding a tape cartridge to the library by creating or updating the tape volume record in the tape configuration database. The cartridge entry process includes the assignment of the cartridge to scratch or private category in the library.

Cartridge System Tape. The base tape cartridge media used with 3480 or 3490 Magnetic Tape Subsystems. Contrast with Enhanced Capacity Cartridge System Tape.

CDS. Control data set.

Cell. A single cartridge location within an Automated Tape Library Dataserver. See also rack number.

CIM. The Common Information Model
An implementation-neutral, object-oriented schema for describing network management information. The Distributed Management Task Force (DMTF) develops and maintains CIM specifications.

Circular file. A type of file that appends data until full. Then, starting at the beginning of the file, subsequent incoming data overwrites the data already there.

Class. A collection of RACF-defined entities (users, groups, and resources) with similar characteristics. Classes are defined in the class descriptor table (CDT), except for the USER, GROUP, and DATASET classes.

Cluster. The combination of a TS7700 Virtualization Engine, 3953 L05 Library Manager, and aTS3500 Tape Library to form a virtual tape subsystem.

Class authority (CLAUTH). An attribute enabling a user to define RACF profiles in a class defined in the class descriptor table. A user can have class authority to zero or more classes.

CLAUTH attribute. See class authority.

Conditional access list. The portion of a resource profile that specifies the users and groups that might access the resource at a specified level when a specified condition is true. For example, with program access to data sets, the condition is that the user must be executing the program specified in the access list. Contrast with standard access list.

Command. A control signal that initiates an action or the start of a sequence of actions.

Command line. On a panel, a display line usually at the bottom of the panel on which only commands can be entered.

Composite Library ID. The library sequence number reported to a host by each attached virtual device.

Concurrent copy. A function to increase the accessibility of data by enabling you to make a consistent backup or copy of data concurrent with the usual application program processing.

Confirmation panel. A DFSMSrmm panel that lets you tell DFSMSrmm to continue or stop a delete or release action. You specify whether or not you want to confirm delete or release requests in your dialog user options.

Container. A receptacle in which one or more exported logical volumes can be stored. A stacked volume containing one or more logical volumes and residing outside a virtual tape server library is considered to be the container for those volumes.

Control data set. A Virtual Storage Access Method (VSAM) key-sequenced data set that contains the complete inventory of your removable media library, as well as the movement and retention policies you define. In the control data set DFSMSrmm records all changes made to the inventory, such as adding or deleting volumes.

Control data set ID. A one-to-eight character identifier for the DFSMSrmm control data set used to ensure that, in a multi-system, multi-complex environment, the correct management functions are performed.

Convenience input. The process of adding a small number of tape cartridges to the IBM 3494 Tape Library Dataserver and IBM 3495 Tape Library Dataserver without interrupting operations, by inserting the cartridges directly into cells in a convenience input station.

Convenience input/output station. A transfer station with combined tape cartridge input and output functions in the IBM 3494 Tape Library Dataservers only.

Convenience input station. A transfer station, used by the operator to add tape cartridges to the IBM 3494 Tape Library Dataserver or an IBM 3495 Tape Library Dataserver, which is accessible from outside the enclosure area.

Convenience output. The process of removing a small number of tape cartridges from the IBM 3494 Tape Library Dataserver or an IBM 3495 Tape Library Dataserver without interrupting operations, by removing the cartridges directly from cells in a convenience input station.

Convenience output station. A transfer station, used by the operator to remove tape cartridges from the Automated Tape Library Dataserver, which is accessible from outside the enclosure area.

Conversion. In DFSMSrmm, the process of moving your removable media library inventory from another media management system to DFSMSrmm. DFSMSrmm manages the inventory and policies once you have converted it.

Create date. Create date for a data set is the date that the data set is written to tape. Create date can also be the date a data set was read if it was created before DFSMSrmm is in use. Create date is updated each time a data set is replaced and not extended. Create date for volumes and other resources defined to DFSMSrmm is the date the resource is defined to DFSMSrmm or the date specified on the command as the create date.

CST. Cartridge System Tape.

D

Data set profile. A profile that provides RACF protection for one or more data sets. The information in the profile can include the data set profile name, profile owner, universal access authority, access list, and other data. See discrete profile and generic profile.

DASD. Direct access storage device.

DASF fast write. An extended function of some models of the IBM 3990 Storage Control in which data is written concurrently to cache and nonvolatile storage and automatically scheduled for destaging to DASD. Both copies are retained in the storage control until the data is completely written to the DASD, providing data integrity equivalent to writing directly to the DASD. Use of DASD fast write for system-managed data sets is controlled by storage class attributes to improve performance. See also dynamic cache management. Contrast with cache fast write.

DASD volume. A DASD space identified by a common label and accessed by a set of related addresses. See also volume, primary storage, migration level 1, migration level 2.

Data column. A vertical arrangement of identical data items, used on list panels to display an attribute, characteristic, or value of one or more objects.

Data entry panel. A panel in which the user communicates with the system by filling in one or more fields.

DCB. Data control block.

Data Facility Sort. An IBM licensed program that is a high-speed data processing utility. DFSORT provides an efficient and flexible way to handle sorting, merging, and copying operations, as well as providing versatile data manipulation at the record, field, and bit level.

Device. This term is used interchangeably with unit. You mount a tape on a unit or device, such as a 3490.

DFSMS environment. An environment that helps automate and centralize the management of storage. This is achieved through a combination of hardware, software, and policies. In the DFSMS environment for MVS, the function is provided by DFSORT, RACF, and the combination of DFSMS/MVS and MVS.

DFSMS/MVS. An IBM S/390® licensed program that provides storage, data, and device management functions. When combined with MVS/ESA SP Version 5 it composes the base MVS/ESA operating environment. DFSMS/MVS consists of DFSMSdfp, DFSMSdss, DFSMShsm, and DFSMSrmm.

DFSMSdfp. A DFSMS/MVS functional component or base element of OS/390, that provides functions for storage management, data management, program management, device management, and distributed data access.

DFSMSdss. A DFSMS/MVS functional component or base element of OS/390, used to copy, move, dump, and restore data sets and volumes.

DFSMShsm. A DFSMS/MVS functional component or base element of OS/390, used for backing up and recovering data, and managing space on volumes in the storage hierarchy.

DFSMShsm-managed volume. (1) A primary storage volume, which is defined to DFSMShsm but which does not belong to a storage group. (2) A volume in a storage group, which is using DFSMShsm automatic dump, migration, or backup services. Contrast with system-managed volume and DFSMSrmm-managed volume.

DFSMShsm-owned volume. A storage volume on which DFSMShsm stores backup versions, dump copies, or migrated data sets.

DFSMSrmm. A DFSMS/MVS functional component or base element of OS/390, that manages removable media.

DFSMSrmm control data set. See control data set.

DFSMSrmm-managed volume. A tape volume that is defined to DFSMSrmm. Contrast with system-managed volume and DFSMShsm-managed volume.

Disaster recovery. A procedure for copying and storing an installation's essential business data in a secure location, and for recovering that data in the event of a catastrophic problem. Compare with vital records.

Discrete profile. A resource profile that provides RACF protection for a single resource.

DISTANT. A DFSMSrmm built-in storage location ID. See built-in storage location.

Distributed Library ID. The unique library sequence number possessed by each cluster in a grid configuration.

DSNB. Data set name block.

Dual copy. A high availability function made possible by nonvolatile storage in some models of the IBM 3990 Storage Control. Dual copy maintains two functionally identical copies of designated DASD volumes in the logical 3990 subsystem, and automatically updates both copies every time a write operation is issued to the dual copy logical volume.

Dump class. A set of characteristics that describes how volume dumps are managed by DFSMShsm.

Duplexing. The process of writing two sets of identical records in order to create a second copy of data.

Dynamic cache management. A function that automatically determines which data sets will be cached based on the 3990 subsystem load, the characteristics of the data set, and the performance requirements defined by the storage administrator.

E

EBCDIC. Extended binary-coded decimal interchange code. A coded character set consisting of 8-bit coded characters.

ECCST. Enhanced Capacity Cartridge System Tape.

EDM. External data manager.

EHPCT. Extended High Performance Cartridge Tape.

EiB. Exbibyte; IEC Standard is 1024 pebibytes.

Eject. The process used to remove a volume from a system-managed library. For an Automated Tape Library Dataserver, the volume is removed from its cell location and moved to the output station. For a manual tape library dataserver, the volume is not moved, but the tape configuration database is updated to show the volume no longer resides in the manual tape library dataserver.

Enhanced Capacity Cartridge System Tape. Cartridge system tape with increased capacity that can only be used with 3490E Magnetic Tape Subsystems. Contrast with Cartridge System Tape.

Entry panel. See data entry panel.

EOV. End of volume.

EREP. Environmental Record Editing and Printing program.

Exa (E). The information-industry meaning depends upon the context:
1. P = 1.152.921.504.606.846.976($2^5$0) for real and virtual storage
2. P = 1,000,000,000.000.000.000 for disk storage capacity (for example, 4 Gb fixed disk)
3. P = 1,000,000,000.000.000.000 for transmission rates.

Expanded output. Expanded output occurs when you specify OUTPUT=FIELDS and EXPAND=YES. For those subcommands for which expanded output applies, your application program receives more variable data than for standard output.

Expiration. The process by which data sets and volumes are identified as available for reuse. In DFSMSrmm, all volumes have an expiration date or retention period set for them either by vital record specification (VRS) policy, by user-specified JCL when writing a data set to the volume, or by an installation default. When a volume reaches its expiration date or retention period, it becomes eligible for release.

Expiration date. The date at which a file is no longer protected against automatic deletion by the system.

Expiration processing. The process of inventory management that ensures expired volumes are released and carries out required release actions on those volumes.

Export. The operation to remove one or more logical volumes from a virtual tape server library. First, the list of logical volumes to export must be written on an export list volume and then, the export operation itself must be initiated.

Export list volume. A virtual tape server logical volume containing the list of logical volumes to export.

Exported logical volume. A logical volume that has one through the export process and now resides on a tacked volume outside a virtual tape server library.

External label. A label attached to the outside of a tape cartridge that is to be stored in an IBM 3494 Tape Library Dataserver or IBM 3495 Tape Library Dataserver. The label might contain the DFSMSrmm rack number of the tape volume.

Extract data set. A data set that you use to generate reports.

F

Field format. Field format is where the output consists of Structured Field Introducers and variable data rather than output in line format.

Filtering. The process of selecting data sets based on specified criteria. These criteria consist of fully or partially-qualified data set names or of certain data set characteristics.

FIPS. Federal Information Processing Standard.

FMID. Function modification identifier.

FRR. Functional recovery routines.

FTP. File transfer protocol.

G

GB. Gigabytes.

GDG. Generation data group.

GDS. Generation data set.

Generation data group. A collection of data sets kept in chronological order. Each data set is a generation data set.

Generation number. The number of a generation within a generation data group. A zero represents the most current generation of the group, a negative integer (-1) represents an older generation and, a positive integer (+1) represents a new generation that has not yet been cataloged.

Generic profile. A resource profile that can provide RACF protection for zero or more resources. The resources protected by a generic profile have similar names and identical security requirements, though with RACFVARS, a generic profile can protect resources with dissimilar names, too. For example, a generic data set profile can protect one or more data sets.

GiB. Gibibyte; IEC Standard is 1024 mebibytes.

Giga (G). The information-industry meaning depends upon the context:
1. G = 1,073,741,824(2^30) for real and virtual storage
2. G = 1,000,000,000 for disk storage capacity (for example, 4 Gb fixed disk)
3. G = 1,000,000,000 for transmission rates.

GMT. Greenwich Mean Time.

GPR. General purpose register.

Grid. A series of clusters connected to one another by means of TCP/IP to form a disaster recovery solution where logical volume attributes and data are replicated across the clusters to ensure the continuation of production work

GRS. Global Resource Serialization.

Guaranteed space. A storage class attribute indicating the space is to be preallocated when a data set is created. If you specify explicit volume serial numbers, the storage management subsystem (SMS) honors them. If space to satisfy the allocation is not available on the user-specified volumes, the allocation fails.

H

Hardware configuration definition. An interactive interface in MVS that enables an installation to define hardware configurations from a single point of control.

HCD. See Hardware configuration definition.

HLQ. High-level qualifier.

HPCT. High Performance Cartridge Tape.

High capacity input station. A transfer station, used by the operator to add tape cartridges to the IBM 3494 Tape Library Dataserver or IBM 3495 Tape Library Dataserver, which is inside the enclosure area.

High capacity output station. A transfer station, used by the operator to remove tape cartridges from the Automated Tape Library Dataserver, which is inside the enclosure area.

Home. See home location.

Home location. For DFSMSrmm, the place where DFSMSrmm normally returns a volume when the volume is no longer retained by vital records processing.

I

IBM. International Business Machines.

ICETOOL. DFSORT's multipurpose data processing and reporting utility.

ICF. Integrated catalog facility.

ICF catalog. A catalog that is composed of a basic catalog structure (BCS) and its elated volume tables of contents (VTOCs) and VSAM volume data sets (VVDSs). See also basic catalog structure and VSAM volume data set.

ID. Identifier. See Identifier (ID).

Identifier (ID). In programming languages, a lexical unit that names a language object; for example, the names of variables, arrays, records, labels, or procedures. An identifier usually consists of a letter optionally followed by letters, digits, or other characters. One or more characters used to identify or name a data element and possibly to indicate certain properties of that data element.
A sequence of bits or characters that identifies a program, device, or system to another program, device, or system.

Import. The operation to enter previously exported logical volumes residing on a stacked volume into a virtual tape server library. First, the list of logical volumes to import must be written on an import list volume and the stacked volumes must be entered, and then, the import operation itself must be initiated.

Import list volume. A virtual tape server logical volume containing the list of logical volumes to import. This list can contain individual logical volumes to import and it can contain a list of stacked volumes in which all logical volumes on the stacked volume are imported.

Imported logical volume. An exported logical volume that has gone through the import process and can be referenced as a tape volume within a virtual tape server library. An imported logical volume originates from a stacked volume that went through the export process.

Improved data recording capability. A recording mode that can increase the effective cartridge data capacity and the effective data rate when enabled and used. IDRC is always enabled on the 3490E Magnetic Tape Subsystem.

Installation defined storage location. A storage location defined using the LOCDEF command in the EDGRMMxx parmlib member.

Interactive Storage Management Facility (ISMF). The interactive interface of DFSMS/MVS that allows users and storage administrators access to the storage management functions.

Interactive System Productivity Facility (ISPF). An IBM licensed program used to develop, test, and run interactive, panel-driven dialogs.

Interface. A shared boundary. Examples include a hardware component to link two devices or a portion of storage or registers accessed by two or more computer programs.

In transit. A volume is in transit when it must be moved from one location to another and DFSMSrmm believes that the move has started, but has not yet received confirmation that the move is complete. For a volume moving from a system-managed library, the move starts when the volume is ejected.

Internal label. The internal label for standard label tapes is recorded in the VOL1 header label, magnetically recorded on the tape media.

Inventory management. The regular tasks that need to be performed to maintain the control data set. See also expiration processing, storage location management processing, and vital record processing.

IPCS. Interactive Problem Control System.

IPL. Initial program load. The process that loads the system programs from the system auxiliary storage, checks the system hardware, and prepares the system for user operations.

ISMF. Interactive Storage Management Facility. An Interactive System Productivity Facility (ISPF) application that provides an interactive set of space management functions for users and storage administrators.

ISPF. Interactive System Productivity Facility.

ISO. International Organization of Standardization.

ITSO. International Technical Support Organization.

IVP. Installation verification procedure.

J

JCL. Job control language. A command language that is used to identify a job to an operating system and to describe the job's requirements.

JES2. Job entry subsystem 2.

JES3. Job entry subsystem 3.

JFCB. Job file control block.

Journal. A sequential data set that contains a chronological record of changes made to the DFSMSrmm control data set. You use the journal when you need to reconstruct the DFSMSrmm control data set.

K

Keyword. A predefined word that is used as an identifier.

KiB. Kibibyte.

Kilo (K). The information-industry meaning depends upon the context:
1. K = 1024(2^10) for real and virtual storage
2. K = 1000 for disk storage capacity (for example, 4000 KB fixed disk)
3. K = 1000 for transmission rates

KSDS. Key-sequenced data set.

L

LCS. Library Control System.

Library Control System. The Object Access Method component that controls optical and tape library operations and maintains configuration information.

Line format. Line format is where text and variable data are formatted into lines suitable for displaying at a terminal or printing on hardcopy output.

LM. Library manager.

LOCAL. A DFSMSrmm built-in storage location ID. See built-in storage location.

Location name. A name given to a place for removable media that DFSMSrmm manages. A location name can be the name of a system-managed library, a storage location name, or the location SHELF, identifying shelf space outside a system-managed library or storage locations.

Logical partition. A subset of a single server that contains resources (processors, memory, and input/output devices). A logical partition operates as an independent system. If hardware requirements are met, multiple logical partitions can exist within a system.

Logical volume. Logical volumes have a many-to-one association with physical tape media and are used indirectly by MVS applications. They reside in a Virtual Tape Server or on exported stacked volumes. Applications can access the data on these volumes only when they reside in a Virtual Tape Server which makes the data available via its tape volume cache or after the data has been copied to a physical volume through the use of special utilities.

LPAR. A function that enables the creation of logical partitions. See logical partition.

Low-on-scratch management. The process by which DFSMSrmm replenishes scratch volumes in a system-managed library when it detects that there are not enough available scratch volumes.

LPA. Link pack area.

LSR. Local shared resources.

M

Management class. A collection of management attributes, defined by the storage administrator, used to control the release of allocated but unused space: to control the retention, migration, and backup of data sets: to control the retention and backup of aggregate groups, and to control the retention, backup, and class transition of objects. If assigned by ACS routine to system-managed tape volumes, can be used to identify a DFSMSrmm vital record specification (VRS).

Management value. See vital record specification management value.

Manual cartridge entry processing. The process by which a volume is added to the tape configuration database when it is added to a manual tape library dataserver. DFSMSrmm can initiate this process.

Manual mode. An operational mode where DFSMSrmm runs without recording volume usage or validating volumes. The DFSMSrmm TSO commands, ISPF dialog, and inventory management functions are all available in manual mode.

Manual tape library. A set of tape drives defined as a logical unit by the installation together with the set of system-managed volumes which can be mounted on those drives. See also tape library. Contrast with automated tape library.

Master system. The MVS system where the master DFSMSrmm control data set resides.

Master volume. A private volume that contains data that is available for write processing based on the DFSMSrmm EDGRMMxx parmlib MASTEROVERWRITE operand.

Media format. The type of volume, recording format and techniques used to create the data on the volume.

Media library. See removable media library.

Media management system. A program that helps you manage removable media. DFSMSrmm is a media management system.

Media name. An up to 8 character value that describes the shape or type of removable media stored in a storage location. Examples of media name are: SQUARE, ROUND, CARTRDGE, or 3480.

Media type. A value that specifies the volume's media type. Media type can be specified as: *, CST, ECCST, HPCT or EHPCT.

MEDIA 1. Cartridge System Tape.

MEDIA 2. Enhanced Cartridge System Tape.

MEDIA 3. High Performance Cartridge Tape.

MEDIA 4. Extended High Performance Cartridge Tape.

Mega (M). The information-industry meaning depends upon the context:
1. M = 1,048,576(2^20) for real and virtual storage
2. M = 1,000,000 for disk storage capacity (for example, 4000 MB fixed disk)
3. M = 1,000,000 for transmission rates

MiB. Mebibyte.

Migration. The process of moving unused data to lower cost storage in order to make space for high-availability data. If you want to use the data set, it must be recalled. See also migration level 1and migration level 2.

Migration level 1. DFSMShsm-owned DASD volumes that contain data sets migrated from primary storage volumes. The data can be compressed. See also storage hierarchy. Contrast with primary storage and migration level 2.

Migration level 2. DFSMShsm-owned tape or DASD volumes that contain data sets migrated from primary storage volumes or from migration level 1 volumes. The data can be compressed. See also storage hierarchy. Contrast with primary storage and migration level 1.

Mount. A host command to load a tape cartridge into a tape drive. A virtual action in which a logical volume is assigned to a virtual tape drive.

MVS. Multiple virtual storage.

MVS image. A single occurrence of the MVS/ESA operating system that has the ability to process work.

N

Name vital record specification. A vital record specification (VRS) used to define additional retention and movement policy information for data sets or volumes.

NEXTVRS. An RMM ADDVRS TSO subcommand operand. See Using NEXT.

NL. No label.

Non-scratch volume. A volume that is not scratch, which means it has valid or unexpired data on it. Contrast with scratch.

NSL. Non-standard label.

O

OAM. Object Access Method.

Object. A named byte stream having no specific format or record orientation.

Object Access Method (OAM). An access method that provides storage, retrieval, and storage hierarchy management for objects and provides storage and retrieval management for tape volumes contained in system-managed libraries.

OCDS. Offline control data set.

Optical disk. A disk that uses laser technology for data storage and retrieval.

Optical volume. Storage space on an optical disk, identified by a volume label. See also volume.

Option line. See command line.

Owner. In DFSMSrmm, a person or group of persons defined as a DFSMSrmm user owning volumes. An owner is defined to DFSMSrmm through an owner ID.

Owner ID. In DFSMSrmm, an identifier for DFSMSrmm users who own volumes.

P

Parallel running. During conversion, when you install DFSMSrmm concurrently with an existing media management system, it is called running in parallel.

Parameter. A variable that is given a constant value for a specified application and that might denote the application.

Partitioned data set (PDS). A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

PDS. Partitioned data set.

Permanent data set. A user-named data set that is normally retained for longer than the duration of a job or interactive session. Contrast with temporary data set.

Peta (P). The information-industry meaning depends upon the context:
1. P = 1,125,899,906,842,624(2^40) for real and virtual storage
2. P = 1,000,000,000,000,000 for disk storage capacity (for example, 4 Gb fixed disk)
3. P = 1,000,000,000,000,000 for transmission rates.

PF. Program function key.

Physical stacked volume. See stacked volume.

Physical volume. Physical volumes have a one-to-one association with physical tape media and are used directly by MVS applications. They might reside in an Automated Tape Library Dataserver or be kept on shelf storage either at vault sites or within the data center where they can be mounted on stand-alone tape drives.

PiB. Pebibyte.

Pool. A group of shelf locations in the removable media library whose rack numbers share a common prefix. The shelf locations are logically grouped so that the volumes stored there are easier to find and use.

Pooling. The process of arranging shelf locations in the removable media library into logical groups.

Pool ID. The identifier for a pool. You define pool IDs in parmlib member EDGRMMxx.

Pool storage group. A type of storage group that contains system-managed DASD volumes. Pool storage groups allow groups of volumes to be managed as a single entity. See also storage group.

PPT. Program properties table.

Primary space allocation. Amount of space requested by a user for a data set when it is created. Contrast with secondary space allocation.

Primary storage. A DASD volume available to users for data allocation. The volumes in primary storage are called primary volumes. See also storage hierarchy. Contrast with migration level 1and migration level 2.

Private tape volume. A volume assigned to specific individuals or functions.

Primary vital record specification. The first retention and movement policy that DFSMSrmm matches to a data set and volume used for disaster recovery and vital record purposes. See also vital record specification and secondary vital record specification.

Profile. Data that describes the significant characteristics of a user, a group of users, or one or more resources. A profile contains a base segment, and optionally, a number of other segments.

Protect mode. In protect mode, DFSMSrmm validates all volume requests.

Pseudo-generation data group. A collection of data sets, using the same data set name pattern, to be managed like a generation data group. The ~ masking character is used in DFSMSrmm to identify the characters in the pattern that change with each generation.

PSW. Program status word.

PTF. Program temporary fix.

Pull list. A list of scratch volumes to be pulled from the library for use.

PUT. Program update tape.

R

RACF. See Resource Access Control Facility.

Rack number. A six-character identifier that corresponds to a specific volume's shelf location in the installation's removable media library, and is the identifier used on the external label of the volume to identify it. The rack number identifies the pool and the external volume serial number for a volume residing in an Automated Tape Library Dataserver. The rack number identifies the pool, the external volume serial, and shelf location number for a volume not residing in an Automated Tape Library Dataserver. The rack number is not written by the tape drive. It exists as an entry in the DFSMSrmm control data set and on the external label of the tape. See also shelf location.

Rack pool. A group of shelves that contains volumes that are generally read-only.

RDS. Retention data set.

Ready to scratch. This describes the condition where a volume is eligible for scratch processing while it resides in a storage location. Since no other release actions are required, the volume can be returned to scratch directly from the storage location.

Recall. When a logical volume is moved from a physical volume to the cache and becomes a virtual volume.

Record-only mode. The operating mode where DFSMSrmm records information about volumes as you use them, but does not validate or reject volumes.

Recording format. For a tape volume, the format of the data on the tape; for example, 18 tracks, 36 tracks, 128 tracks or 256 tracks.

Recovery. The process of rebuilding data after it has been damaged or destroyed, often by using a backup copy of the data or by reapplying transactions recorded in a journal.

Relative start generation. Relative generation zero is the latest generation of a tape; Relative generation -1 is the previous generation of that tape. Relative generation -2 is the generation before the previous one.

REMOTE. A DFSMSrmm built-in storage location ID. See built-in storage location.

Removable media. See volume.

Removable media library. The volumes that are available for immediate use, and the shelves where they might reside.

Resource Access Control Facility (RACF). An IBM licensed program that provides for access control by identifying and verifying the users to the system; authorizing access to protected resources; logging the detected unauthorized attempts to enter the system; and logging the detected accesses to protected resources.

Resource Group. A collection of structured fields that describe the attributes of a resource such as a volume.

Restructured Extended Executor (REXX) Language. A general-purpose, high-level programming language, particularly suitable for EXEC procedures or programs for personal computing.

Retention date. Retention date can be the date that a data set or volume is retained by a vital record specification (VRS) or the date of the inventory management run when the data set or volume is no longer retained by a VRS.

Retention period. The time for which DFSMSrmm retains a volume or data set before considering it for release. You can retain a data set or volume as part of disaster recovery or vital records management. You set a retention period through a vital record specification (VRS) that overrides a data set's expiration date.

Retention type. The types of retention for which DFSMSrmm retains a volume or data set before considering it for release. The retention types for data sets are BYDAYSCYCLE, CYCLES, DAYS, EXTRADAYS, LASTREFERENCEDAYS, UNTILEXPIRED, and WHILECATALOG. The retention types for volumes are DAYS and CYCLE.

REXX. Restructured Extended Executor Language.

IBM RMF™. IBM Resource Measurement Facility™.

RMM. Removable Media Manager.

RMM complex (RMMplex). One or more MVS images that share a common DFSMSrmm control data set.

RMODE. Residence mode.

R/W. Read/write.

S

SAF. System Authorization Facility.

Scratch. The status of a tape volume that is available for general use, because the data on it is incorrect or is no longer needed. You request a scratch volume when you omit the volume serial number on a request for a tape volume mount.

Scratch allocation assistance (SAA). An extension of the device allocation assistance (DAA) function for scratch mount requests. It filters the list of clusters in a grid to return to the host a smaller list of candidate clusters specifically designated as scratch mount candidates.

Scratch pool. The collection of tape volumes from which requests for scratch tapes can be satisfied. Contrast with rack pool.

Scratch processing. The process for returning a volume to scratch status once it is no longer in use and has no outstanding release actions pending.

Scratch tape. See scratch volume.

Scratch volume. A tape volume that contains expired data only. See scratch.

SDB. Structured database.

SDSF. Spool Display and Search Facility.

Secondary space allocation. Amount of additional space requested by the user for a data set when primary space is full. Contrast with primary space allocation.

Secondary vital record specification. The second retention and movement policy that DFSMSrmm matches to a data set and volume used for disaster recovery and vital records purposes. See also vital record specification and primary vital record specification.

Selective device access control (SDAC). A function that allows only certain logical control units or subsystem ids within a composite library to have exclusive access to one or more VOLSER ranges of volumes for host-initiated mounts, ejects, and attributes or categories changes.

SFI. Structured field introducer.

Shelf. A place for storing removable media, such as tape and optical volumes, when they are not being written to or read.

Shelf location. A single space on a shelf for storage of removable media. DFSMSrmm defines a shelf location in the removable media library by a rack number, and a shelf location in a storage location by a bin number. See also rack number and bin number.

Shelf-management. Is the function provided to manage the placement of volumes in individual slots in a location. Shelf-management is provided for the removable media library using rack numbers. For storage locations it is optional as defined by the LOCDEF options in parmlib and uses bin numbers.

Shelf-resident volume. A volume that resides in a non-system-managed tape library.

Shelf space. See shelf.

SL. Standard label.

Slot. See shelf location.

SMF. System Management Facility.

SMP/E. System Modification Program Extended.

SMS. Storage management subsystem.

SMSplex. System-managed storage complex. A group of storage devices managed collectively by a single operating system. The system automates storage management through classification of data sets and objects.

Stacked volume. Stacked volumes have a one-to-one association with physical tape media and are used in a Virtual Tape Server to store logical volumes. Stacked volumes are not used by MVS applications but by the Virtual Tape Server and its associated utilities. They can be removed from a Virtual Tape Server to allow transportation of logical volumes to a vault or to another Virtual Tape Server.

Standard access list. The portion of a resource profile that specifies the users and groups that can access the resource and the level of access granted to each.

Standard label. An IBM standard tape label.

Standard output. Standard output is the amount of variable data displayed, printed or put into a REXX variable in response to a subcommand. When you specify OUTPUT=LINES or EXPAND=NO with OUTPUT=FIELDS, your application program receives standard output as opposed to expanded output.

Storage administrator. A person in the data processing center who is responsible for defining, implementing, and maintaining storage management policies.

Storage class. A collection of storage attributes that identify performance goals and availability requirements, defined by the storage administrator, used to select a device that can meet those goals and requirements.

Storage group. A collection of storage volumes and attributes, defined by the storage administrator. The collections can be a group of DASD volumes or tape volumes, or a group of DASD volumes and optical volumes treated as a single object storage hierarchy.

Storage location. A location physically separate from the removable media library where volumes are stored for disaster recovery, backup, and vital records management.

(Storage) location dominance. The priority used by DFSMSrmm to decide where to move a volume within the removable media library during vital record specification (VRS) processing. It covers all the locations; SHELF, storage locations, and system-managed tape libraries.

Storage location management processing. The process of inventory management that assigns a shelf location to volumes that have moved as a result of vital record processing. See also vital record processing.

Stripe. In DFSMS/MVS, the portion of a striped data set that resides on one volume. The records in that portion are not always logically consecutive. The system distributes records among the stripes such that the volumes can be read from or written to simultaneously to gain better performance. Whether it is striped is not apparent to the application program.

Striping. A software implementation of a disk array that distributes a data set across multiple volumes to improve performance.

Structured field. Output from the DFSMSrmm application programming interface consisting of a Structured Field Introducer and output data.

Structured field introducer. An 8-byte entity that either introduces the beginning of a group of data or introduces output data that immediately follows the introducer.

Subsystem. A special MVS task that provides services and functions to other MVS users. Requests for service are made to the subsystem through a standard MVS facility known as the subsystem interface (SSI). Standard MVS subsystems are the master subsystem and the job entry subsystems JES2 and JES3.

Subsystem interface (SSI). The means by which system routines request services of the master subsystem, a job entry subsystem, or other subsystems defined to the subsystem interface.

SUL. IBM standard and user header or trailer label.

SVC. Supervisor call.

System-managed storage. Storage managed by the storage management subsystem. The SMS attempts to deliver required services for availability, performance, and space to applications. See also DFSMS environment.

System-managed tape library. A collection of tape volumes and tape devices, defined in the tape configuration database. A system-managed tape library can be automated or manual. See also tape library.

System-managed volume. A DASD, optical, or tape volume that belongs to a storage group. Contrast with DFSMShsm-managed volume and DFSMSrmm-managed volume.

System programmer. A programmer who plans, generates, maintains, extends, and controls the use of an operating system and applications with the aim of improving overall productivity of an installation.

T

Tape configuration database (TCDB). One or more volume catalogs used to maintain records of system-managed tape libraries and tape volumes.

Tape librarian. The person who manages the tape library. This person is a specialized storage administrator.

Tape library. A set of equipment and facilities that support an installation's tape environment. This can include tape storage racks, a set of tape drives, and a set of related tape volumes mounted on those drives. See also system-managed tape library and automated tape library.

Tape Library Control System (TLCS). IBM program offering 5785-EAW. DFSMSrmm replaces TLCS.

Tape Library Dataserver. A hardware device that maintains the tape inventory associated with a set of tape drives. An Automated Tape Library Dataserver also manages the mounting, removal, and storage of tapes. An automated tape library that supports system-managed storage of tape volumes. IBM Automated Tape Library Dataservers include the IBM 3494 Tape Library Dataserver and the IBM 3495 Tape Library Dataserver.

Tape storage group. A type of storage group that contains system-managed private tape volumes. The tape storage group definition specifies the system-managed tape libraries that can contain tape volumes. See also storage group.

Tape subsystem. A magnetic tape subsystem consisting of a controller and devices, which allows for the storage of user data on tape cartridges. Examples of tape subsystems include the IBM 3490 and 3490E Magnetic Tape Subsystems.

Tape volume. A tape volume is the recording space on a single tape cartridge or reel. See also volume.

TB. Terabytes.

TCDB. Tape Configuration Database.

TCP/IP. (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). TCP/IP is a two-layer program. The higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, Internet Protocol, handles the address part of each packet so that it gets to the correct destination.

Temporary data set. An uncataloged data set whose name begins with & or &&, that is normally used only for the duration of a job or interactive session. Contrast with permanent data set.

Tera (T). The information-industry meaning depends upon the context:
1. T = 1,099,511,627,776(240) for real and virtual storage
2. T = 1,000,000,000,000 for disk storage capacity (for example, 4 TB of DASD storage)
3. T = 1,000,000,000,000 for transmission rates

TiB. Tebibyte.

TMC. Tape management catalog.

TRS. Tape retention system.

TSO. Time sharing option.

U

Until expired. Allows the use of vital record specification (VRS) policies for managing retention in a location as long as the volume expiration date has not been reached.

Use attribute. (1) The attribute assigned to a DASD volume that controls when the volume can be used to allocate new data sets; use attributes are public, private, and storage. (2) For system-managed tape volumes, use attributes are scratch and private.

User volume. A volume assigned to a user, that can contain any data and can be rewritten as many times as the user wishes until the volume expires.

Using AND. A method for linking DFSMSrmm vital record specifications (VRSs) to create chains of VRSs. DFSMSrmm applies policies in chains using AND only when all the retention criteria are true.

Using NEXT. A method for linking DFSMSrmm vital record specifications (VRSs) to create chains of VRSs. DFSMSrmm applies policies in chains using NEXT one vital record at a time.

Utilities. Utility programs

Utility programs. A computer program used for general support of the processes of a computer; for instance, a diagnostic program.

V

Virtual export. Mark a volume as exported by using the DFSMSrmm subcommands.

Virtual input/output (VIO) storage group. A type of storage group that allocates data sets to paging storage, which simulates a DASD volume. VIO storage groups do not contain any actual DASD volumes. See also storage group.

Virtual tape drives. Emulated tape drives that occur when the TS7740 Server emulates the function and operation of 3490 Enhanced Capacity (3490E) tape drives.

Virtual Tape Server (VTS). This subsystem, integrated into the IBM Magstar® 3494 Tape Library, combines the random access and high performance characteristics of DASD with outboard hierarchical storage management and virtual tape devices and tape volumes.

Virtual volume. Data storage on DDMs that shows the same characteristics to a host application as a physical tape volume and contains data written or read through a virtual tape drive.

Vital record group. A set of data sets with the same name that matches the same DFSMSrmm vital record specification (VRS).

Vital record processing. The process of inventory management that determines which data sets and volumes DFSMSrmm needs to retain and whether a volume needs to move. These volumes and data sets have been assigned a vital record specification (VRS).

Vital record specification (VRS). Policies defined to manage the retention and movement of data sets and volumes used for disaster recovery and vital records purposes.

Vital record specification management value. A one-to-eight character name defined by your installation and used to assign management and retention values to tape data sets. The vital record management value can be any value you chose to create a match between a vital record specification (VRS) and data sets and volumes in your installation. By matching the vital record specifications to the data set or volumes, DFSMSrmm applies the retention and movement policies you define in the VRSs. During inventory management VRSEL processing, DFSMSrmm selects the correct, best matching VRS for a tape data set or volume.

Vital records. A data set or volume maintained for meeting an externally-imposed retention requirement, such as a legal requirement. Compare with disaster recovery.

VMF. Volume master file.

VMS. Volume management system.

VNL. Volume not in library.

VOLSER. Volume serial number.

Volume. The storage space on DASD, tape, or optical devices, which is identified by a volume label. See also DASD volume, logical volume, optical volume, stacked volume, and tape volume.

Volume catalog. See tape configuration database.

Volume expiration date. The date the volume will expire based on the highest expiration date of the data sets that reside on the volume.

Volume serial number (VOLSER). An identification number in a volume label that is assigned when a volume is prepared for use on the system. For standard label volumes, the volume serial number is the VOL1 label of the volume. For no label volumes, the volume serial number is the name the user assigns to the volume. In DFSMSrmm, volume serial numbers do not have to match rack numbers.

VPDD. Vault pattern description data set.

VRS. Vital record specification.

VTS. Virtual Tape Server.

VSAM volume data set (VVDS). A data set that describes the characteristics of VSAM and system-managed data sets residing on a given DASD volume; part of an integrated catalog facility catalog. See also basic catalog structure and integrated catalog facility catalog.

W

warning mode. The operating mode in which DFSMSrmm validates volumes as you use them, but issues warning messages when it discovers errors instead of rejecting volumes.

WORM. Write Once, Read Many. Any storage media that allows data to be written only once per storage unit and never changed for secure archival purposes. It is allowed to add new content that does not change previously written units of storage.

Write-to-operator. An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that might need to be corrected.

WTO. See write-to-operator.

Write-mount count. A 16-bit sister value of WWID (World Wide Identifier) that represents the total number of times an LWORM volume is mounted and modified due to a write operation. This value is seeded to zero for a newly bound LWORM volume before the first write from BOT (beginning of tape).

WWID. World Wide Identifier. A world-unique, 12-byte, 24-hex digit value assigned to a media cartridge that identifies a volume beyond volume serial. This identifier is also referred to as a cartridge unique identifier or (CUID).

Y

Yotta (Y). The information-industry meaning depends upon the context:
1. P = 1,208,925,819,614,629,174,706,176(2^70) for real and virtual storage.
2. P = 1,000,000,000,000,000,000,000,000 for disk storage capacity (for example, 4 Gb fixed disk).
3. P = 1,000,000,000,000,000,000,000,000 for transmission rates.

Z

Zetta (Z). The information-industry meaning depends upon the context:
1. P = 1,180,591,620,717,411,303,424(2^60) for real and virtual storage.
2. P = 1,000,000,000,000,000,000,000 for disk storage capacity (for example, 4 Gb fixed disk).
3. P = 1,000,000,000,000,000,000,000 for transmission rates.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this Redbooks publication.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 686.

- *Converting to Removable Media Manager: A Practical Guide*, SG24-4998
- *Converting to DFSMSrmm from CA-1*, SG24-6241
- *Converting to DFSMSrmm from TLMS*, SG24-6242
- *Converting to DFSMSrmm from Control-T*, SG24-6243
- *Converting to DFSMSrmm from ICF-usercatalog*, SG24-6244
- *Converting to DFSMSrmm from ZARA/AutoMedia*, SG24-6275
- *Merging Systems into a Sysplex*, SG24-6818
- *z/OS V1R3 and V1R5 DFSMS Technical Guide*, SG24-6979
- *z/OS V1R8 DFSMS Technical Update*, SG24-7435-00
- *DFSMS V1.10 and EAV Technical Guide*, SG24-7617-00
- *z/OS V1.11 DFSMS Release Guide*, SG24-7768-00
- *DFSMShsm Fast Replication Technical Guide*, SG24-7069
- *z/OS V1.12 DFSMS Technical Update*, SG24-7895

Other publications

These publications are also relevant as further information sources:

- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS MVS System Management Facilities (SMF)*, SA22-7630
- *z/OS MVS System Messages, Vol. 5 (EDG-GFS)*, SA22-7635
- *z/OS SecureWay Security Server RACF Security Administrator’s Guide*, SA22-7683
- *z/OS TSO/E Customization*, SA22-7783
- *z/OS DFSMSrmm Guide and Reference*, SC26-7404
- *z/OS DFSMSrmm Implementation and Customization Guide*, SC26-7405
- *z/OS DFSMSrmm Reporting*, SC26-7406

Online resources

These web sites are also relevant as further information sources:

- z/OS information wizardry

  http://www-03.ibm.com/systems/z/os/zos/wizards//

- DFSORT product highlights

  http://www-01.ibm.com/support/docview.wss?uid=isg3T7000077

- z/OS DFSMSrmm Customization Wizard

  http://www-03.ibm.com/systems/z/os/zos/downloads/#rmm1

- Mainstar website

  http://mainstar.rocketsoftware.com

How to get Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Numerics

3420 445
3480 cartridge system tape 277, 445
3490 cartridge system tape 277, 445
3494 Automated Tape Library Dataserver 273
3495 Automated Tape Library Dataserver 273
3590 High Performance Tape Subsystem 445
3592 Model E05 14

A

ABEND 386
access method services REPRO command 235
ACF2 DFSMSrmm definitions 659
ACS processing
  SMS read-only variables 446
ACS processing pooling decisions 466
ACS routines 385, 444, 462
  EDGRMMxx OPTION subparameters 442
  EDGUX100 user exit 443
  how it works 443
  MSPOOL variable 444
  PREACS processing 445
  pre-ACS processing 444
  PREACS subparameter 443
  SMSACS subparameter 443
  VRSs in MC routine 466
action records 597
allocation 30
ANDVRS operand 393, 435
API
  See application programming interface
application programming interface 35
ATL
  See automated tape library
automated tape library

B

Basic Tape Library Support 273, 445
  LIBRARY command 274
  scratch pool 274
bin number 42
BronzePlex considerations for RMMplex 614
BTLS
  See Basic Tape Library Support

C

CAS (catalog address space) 330
catalog address space (CAS) 330
catalog interface 32
catalog synchronization 65, 330
  authorization required 332
  catalog search interface 330
  disabling 337
  EDGHSKP utility 341
  enabling 332
  enabling JCL 345, 356
  journal considerations 334
  LISTCONTROL subcommand 337
  message data set 333
  OPTION CATSYSID 330
  synchronize catalogs 352
CA-Top Secret definitions 656
CATSYNCH 244
CATSYSID operand 350
CBRUXCUA exit 32, 625
CBRUXEJC exit 32, 625
CBRUXENT exit 32, 284, 625
CBRUXVNL exit 32, 625
CDS 12
  See control data set
CDS merge 604
CDS split 608
CDSID parameter 200
CLIENT parameter 200
client/server
  catalogs 206
  client start 205
  firewall 204
  LOCALTASKS 194
  operator command 210
  PORT 194
  SERVERNAME 193
  SERVERTASKS 194
  TCP/IP 205
client/server support 18, 66
command mode query 199
commands
  DISPLAY TCPIP 195
  HOMETEST 195
  HOSTNAME 196
  MODIFY DFRMM,Q A 210
  NSLOOKUP 199
  OMVS 196
  QUERY ACTIVE 210
  TSO NETSTAT 195
concurrent copy 235
control data set 24
  action record 597
  backup 215, 235
  backup and recovery 25
  bin record 601
  contents 596
  control record 597
  data set record 597
  maintaining 234
  mend 260
  merge 596

overview 28
owner record 598
product record 599
rack record 599
reorganize 239
split 596
verify 215, 259
volume record 602
VRS record 598
control record 597
CSDID 12

D

data set cycle 383
data set retention 56
data set VRS 383
dataclass 14
deactivate RACF functions 495
DEVSUP command 488
DEVSUPxx keywords
TAPEAUTHDSN 480
TAPEAUTHF1 480
TAPEAUTHRC4 481
TAPEAUTHRC8 481
DEVSUPxx parmlib member 480, 489
DFSMS 1.8
tape data set authorization 492
DFSMS ACS support 414
DFSMSdfp 30
DFSMSdss 31
DFSMSdss concurrent copy 649
DFSMShsm 31
consistency checking 273
TAPESECURITY 495
DFSMShsm tapes 395
TVEXTPURGE 414
DFSMSrmm
error diagnostic messages 22
interfaces 29
special character support 21
structure 27
TSO/E help 22
DFSMSrmm client 67
DFSMSrmm reporting
EDGRPTD utility 218
EDGRRPTE exec 218
DFSMSrmm server 67
disable catalog synchronization 337
DISPLAY SMS command 454
disposition processing 250
duplicate data set names 618
duplicate volsers 615, 618, 621

E

EDGBKUP utility 237
EDGDFHSM program interface 31
EDGHSKP 12
EDGHSKP utility 212, 234
VERIFY operand 430
EDGINERS 276
EDGINERS utility 245
EDGMSGEX exit 30
EDGRMMxx 13, 497
EDGRMMxx member 617, 622
EDGRMMxx PARMLIB member
ACS processing 464
EDGRMMxx parmlib member 13
EDGRPTD utility 218
EDGRRPTE 13–14
EDGRRPTE exec 218
EDGSPLCS utility 264
EDGTVEXT program interface 31
EDGUTIL 12
EDGUTIL program 617, 619–620
EDGUTIL utility 203, 243, 259, 627
EDGUX100 exit 414, 625
EDGUX100 user exit 443
EDGUX200 exit 625
enabling catalog synchronization 332
enhanced multilevel security 18
Enterprise Encrypted Format 2 (EEFMT2) 13
Enterprise Format 1 (EFMT1) 13
Enterprise Format 2 (EFMT2) 13
entry 49
exits
CBRUXCUA 32
CBRUXEJC 32
CBRUXENT 32
CBRUXVNL 32, 63
EDGMSGEX 30
EDGUX100 414
IGDACSXT 415
IGXMSGEX 30
expiration date decisions 460
expiration processing 286
expiration processing HOUSEKEEP function 653

F

fast tape positioning 32
foreign tapes 252
foreign volumes 50

G

GDG
See generation data group
generation data group 385
global confirmation 219
GoldPlex considerations for RMMplex 614

H

high-speed locate 32
home location 42, 56, 59
HOUSEKEEP function 24
HOUSEKEEP processing 208
housekeeping 212, 215, 383
HSC/MVS 276

I
ICH408I message 481, 500
ICHBLP resource 479
ICHRIN03 635
ichrin2 635
IDCAMS REPRO 604
IEBGENER utility 146
IGDACSXT exit 415
IGXMSGEX exit 30
init 49
interfaces
  short-on-scratch processing 32
inventory management 212, 275
  activities 25
  CDS backup 215
  CDS verify 215
  daily applications 214
  expiration processing 216
  EXPROC 286
  housekeeping 215
  monthly applications 214
  open data sets 220
  permanent I/O errors 219
  preparation 215
  report extract 217
  scheduling tasks 213
  storage location management 216
  trial run 217
  vital records processing 216
  weekly applications 214
IPL considerations 129
ISPF dialog 33

J
journal
  activate 344, 355
  disable 334
journal data set 28
journal percentage full threshold 652

L
LASTREFERENCEDAYS 387
LCLTCTC001I 425–426, 499
library management 43, 60
library name 38
LISTCONTROL subcommand 337
loan location 36

M
management class 385
management value 384
manual tape library
master volume 49
media types 13
MEND parameter 260
merging RMMplexes 614
message data set 333
messages
  CBR3006I 451
  EDG0105I 205, 334, 342, 352
  EDG0122I 334
  EDG0204I 334
  EDG0358D 206
  EDG2235E 330
  EDG2236I 22, 330
  EDG2237E 330
  EDG6101E 209
  EDG6137E 208
  IDC3009I 500
  IDG008I 459, 461
migration considerations 466
MODIFY DFRMM command 464
movement 60
movement policies 56
  DELAY 392
  LOCATION 392
  PRIORITY 392
  STORENUMBER 392
MSPOOL variable 444
MTL
  See manual tape library
multi-volume chain 146
multivolume set 375
MVS JCL DSNAME 13
MVS MODIFY command 205

N
name VRS 388
NEXTVRS operand 393
non-existent volumes 257
non-IBM libraries 275
non-system managed volumes 444
non-system-managed tape libraries 445
non-system-managed tape library 42
NOSECLEVEL 13

O
OAM
  See Object Access Method
Object Access Method 32, 454
OCE
  See open/close/end of volume
OPEN 387
Open MVS console commands 196
open/close/end of volume 30
OpenPegasus CIMOM 14
operating DFSMSrmm
  initialize and erase volumes 245
  processing labels 250
  restart 205
  start 204
OPTION command
  PREACS 443
  SMSACS 443
  SYSID 256

P

PARMLIB members 72
PARMLIB options
  MOVEBY(SET) 59
  OPTION RETAINBY 375
  OPTION TVEXTPURGE 414
  OPTION VRSEL(NEW) 389
  OPTION VRSEL(OLD) 389
  REJECT 62
  VLPOOL 252, 255
  VRSJOBNAME 378
partition an SMS library 283
partitioning a library 285
pending release 49
percentage full threshold 652
PlatinumPlex considerations for RMMPlex 614
policy management
  housekeeping 383
  matching data sets 385
  matching volumes 388
  overview 369
  retention types 390–391
  VRS modification 415
  VRS parameters 389
  VRS types 383
pooling types 255
pools 255
PREACS processing 445
pre-ACS processing 444
  benefits 468
pre-ACS support 415, 467
protect tape volumes 483

R

RACF 31
  assigning a user ID 634
  authorize ABARS to DFSMSrmm resources 651
  authorize DFSMShsm to DFSMSrmm resources 649
  authorizing user groups 639
  automatic tape security support 485
  considerations for RMM 625
  defining ABARS procedure names 650
  defining DFSMShsm procedure names 649
  DFSMShsm and DFSMSrmm considerations 486
  DFSMSrmm access list 27
  DFSMSrmm resource profiles 644
  discrete profiles 642
  discrete TAPEVOL profile 485
  EDGBKUP procedure 652
  EDGLABEL procedure 654
  EDGXPROC procedure 653, 655
  enhanced generic naming considerations 642
  FACILITY class profiles 636
  multisystem considerations 649, 651
  protecting DFSMSdss resources 649
  refresh instorage profiles 487
  STARTED class 635
  started procedures table 635
  started task user ID 634
  TAPEDSN class 483
  TAPEVOL class 483
  TAPEVOL profiles 485–486
  TPRACF operand 485
  TVTOC 485
  user groups 122, 638
  VLPOOL command 485–486
RACF option TAPEDSN 479
RACF profiles 486
RACF tape security support 484
RACF TAPEVOL 492
rack number 37
rack pool 254
Redbooks Web site 686
REJECT 13
REJECT command 62
release functions 1
release options
  EXPIRYDATEIGNORE 393
  SCRATCHIMMEDIATE 393
removable media library 36
REPORT 17 13
REPORT17 14
reports
  searches and lists 26
RETAINBY 375
retention limits
  COUNT 391
  DELETEDATE 392
retention policies 56
retention types
  BYDAYSCYCLE 390
  CYCLES 390
  DAYS 390
  EXTRADAYS 391
  LASTREFERENCEDAYS 390
  UNTILEXPIRED 390
  WHILECATALOG 390
rmcsh3a 638–639
rmcsh3c 642
rmcsh3f 649
rmcsh3g 649
rmcsh3h 650
rmcsh3i 652
rmcsh3j 653, 655
rmcsh3k 654
rmcsh3l 487
RMM ADDVOLUME subcommand 465, 473
RMM ADDVRS subcommand 385
RMM CHANGEVOLUME subcommand 465
RMM CIM provider 14
RMM OPTION command, TPRACF(N) 22
RMM SEACHVOLUME subcommand 497
RMM SEARCHVOLUME subcommand 275
RMM TSO command
  from the console 16, 34
RMMplex 66
  action records 618
  backing out the merge 628
  backing up CDS and journals 626

bin numbers record 621
catalog synchronization 628
CDS and Journal data set sizes 624
CDS record descriptions 615
CDSID value 617
considerations for BronzePlex 614
considerations for GoldPlex 614
considerations for PlatinumPlex 614
control record 617
data set record 618
description of merge job steps 626
DFSMSrmm exits 625
duplicate data set names 618
duplicate volsers 615, 618, 621
EDGRMMxx member 617, 622
EDGUTIL program 617, 619–620
handling duplicate CDs records 616
HOUSEKEEP processing 208
LOCDEF keyword considerations 623
managing volumes 206
merge methodology 625
move records 618
OPTIONS keyword considerations 622
overview 192
owner record 618
PARMLIB options 622
processing record types in merge job 626
product record 619
RACF considerations 207
rack numbers record 620
REJECT keyword considerations 623
relationship to TCDB 624
role of DEVSUPxx member 624
running the EDGUTIL utility after the merge 627
sample backup JCL 629
sample catalog synchronization job 631
sample merge JCL 629
security definitions 625
sharing the started task JCL 623
system-managed libraries 208
toleration service 615
useful documentation 632
VLPOOL keyword considerations 623
VRS record 618
RMMSCR procedure 32
rmmxml.xsd 18
rsch3d 634

S

SAF tape security 16, 476
SAMPLIB members
EDGCVRSX 439
EDGJACTP 430
EDGJBKP1 215
EDGJBKP2 215
EDGJCMOV 219
EDGJDHKP 217
EDGJEJC 218
EDGJEXP 217
EDGJLOPC 212
EDGJMOVE 218
EDGJSCRL 217
EDGJSRDS 439
EDGJVFY 215
EDGJWHKP 217
EDGLABEL 245
EDGPHKPA 215
scratch pooling 255, 257
scratch volume 49
security 27
security classification 27
SECURITY CONFLICT status 500
security policies 633
SERVER parameter 201
SETROPTS CLASSACT(TAPEVOL) 479
SETROPTS NOCLASSACT(TAPEVOL) 479
SETROPTS NOTAPEDSN 479
SETROPTS TAPEDSN 479
SETSMS command 459
shelf location 37
SLSUX06 exit 278
SLUADMIN utility 276
SLUCONDB program 277
SLUDRRMM routine 277
SMA ACS support &MSPARM 467
SMS
&ACCT_JOB 447
&ACCT_STEP 447
&ACSENVIR 447
&DSORG 447
&DSTYPE 448
&EXPDT 448
&FILENUM 448
&GROUP 448
&HLQ 448
&JOB 448
&LABEL 449
&LIBNAME 449
&LLQ 449
&MSPARM 449
&MSPDEST 449
&MSPOLICY 449, 468
&MSPOOL 449, 468
&NQUAL 450
&PGM 450
&SYSNAME 450
&SYSPLEX 450
&UNIT 450
&USER 450
&XMODE 450
read-only variables 446
SMS ACS processing VRS definitions 466
SMS ACS routines 442
SMS ACS support 60
&ACSENVIR variable 446, 467
&MSDEST variable 467
&MSPOLICY variable 467
&MSPOOL variable 467
EDGUX100 exit 443, 466–467
expiration date decisions 460

how it works 443
IGDACSXT exit 467
implementing 451
management class define 460
MSPOLICY 60
MSPOOL 60
PREACS 443
pre-ACS interface 467
read only variables 446
scratch pooling 443, 466
SMS read-only variables 446
SMSACS 443
tape library define 451
tape storage group define 455
SMS ACS support tailoring ACS routines 462
SMS environment 451
SMS interface
SMS policy 446
SMS read-only variables 446
SMSplex relationship to RMM merge 624
software interfaces 29
software products 251
spltc 596
spltcnv 613
spltm 604
splts 608
SSI
see Subsystem Interface
stacked volumes 470
stacked volumes support 469
non-VTS 473
sticky labels 250
STK tape libraries 275
storage location 36
built-in 42
installation-defined 42
storage location management 43, 60, 275
strtpx 634
subsystem interface 27
system-managed tape
library name 38
system-managed tape libraries 61
partitioning 285
system-managed tape library 38

T

tape configuration database 38
tape configuration database (TCDB) 263
tape data set authorization
implementation 488
tape data set security 27, 475, 481
tape data sets record processing 50
tape drive cartridge loader 463
tape library define panel 452
tape security
ARCCMDxx parmlib member 495
DEVSUPnn parmlib member 489
DEVSUPxx parmlib member 476
error messages 499
ICHBLP resource 480
implementtion 488
nonstandard label tapes 479
SAF 476
TAPEVOL and TAPEDSN processing 477
using combined functions 476
tape security support 484
tape volume catalog 624
TAPEAUTHRC4(ALLOW) 476
TAPEAUTHRC8(WARN) 476
TAPEDSN option 27, 475, 477
TAPEDSN RACF class 483
TAPEVOL class 27, 475, 477
TAPEVOL class profile 479
TAPEVOL profile
TVTOC 478
TAPEVOL profiles 497
TAPEVOL RACF class 483
target destination 59
TCDB
See tape configuration database
TCP/IP 35, 66
Tivoli OPC 212
applications 213
batch loader 212
calendars 212
ejecting volumes 218
erasing and labeling volumes 217
Event-Triggered Tracking 213
move confirmation 219
operation 213
producing reports 218
scratch processing 217
scratch reporting 217
special resources 213
workstations 212
Tivoli Storage Manager 31
TPRACF operand 485
TPRACF OPTION 496
TPRACF(AUTOMATIC) 496
TPRACF(CLEANUP) 496
TPRACF(CLEANUP) option 476
TPRACF(PREDEFINED) 496
TSM
See Tivoli Storage Manager
TSO commands 34
TSO RMM subcommands
LISTCONTROL OPTION 335
TVEXTPURGE 414

U

user address space 28
user volume 49
utilities
EDGHSKP 25
UXTABLE 439

V

validation 51
vault 36

VERIFY(SMSTAPE) 259
VERIFY(VOLCAT) 259
Virtual Tape Server 469
    export processing 471
    import processing 471
virtual tape solutions 469
vital record specification
    defining 465
    delete 415
    management value 384
    primary VRS 384
    secondary VRS 384
    trial run 430
vital record specification processing 370
vital record specifications 56
VLPOOL 13
VLPOOL command 255, 497
    SYSID 256
VLPOOL RACF 496
VOLCAT 38
volume
    entry 49
    erase 217
    init 49
    initialize 217
    life cycle 50
    logical 49
    management 48
    master 49
    pending release 32, 49
    physical 48
    scratch 49
    shelf resident 42
    stacked 49
    status 49
    user 49
volume retention 59
volume serial number 388
volume VRS 388
VRS
    See vital record specification
    See vital record specifications
VRS chain 388, 393
    ANDVRS 393
    NEXTVRS 393
VRS management class 58
VRS management value 58, 386
VRS parameters 389
VRS processing
    CATRETPD 371
    MOVEBY 374
    RETAINBY 375
    REUSEBIN 377
    VRSCHANGE 377
    VRSEL 379
    VRSJOBNAME 378
    VRSMIN 379
VRS subchain 393
VRS types 383
    data set 383
    name 388
    volume 388
VRSEL(NEW) 384
VTS
    See Virtual Tape Server
VTS stacked volume support 470

W

WHILECATALOG 386
WORM tapes 18

Z

z/OS V1R3 enhancements 19–20
z/OS V1R5 enhancements 18
z/OS V1R6 enhancements 2, 4–5, 8, 12, 15–17

Redbooks

DFSMSrmm Primer

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

DFSMSrmm Primer

Learn how DFSMSrmm manages your tape environment

Discover the latest DFSMSrmm enhancements

Find DFSMSrmm implementation details

DFSMSrmm from IBM is the full function tape management system available in IBM OS/390 and IBM z/OS. With DFSMSrmm, you can manage all types of tape media at the shelf, volume, and data set level, simplifying the tasks of your tape librarian.

Are you a new DFSMSrmm user? Then, this IBM Redbooks publication introduces you to the DFSMSrmm basic concepts and functions. You learn how to manage your tape environment by implementing the DFSMSrmm management policies.

Are you already using DFSMSrmm? In that case, this publication provides the most up-to-date information about the new functions and enhancements introduced with the latest release of DFSMSrmm. You will find useful information for implementing these new functions and getting more benefits from DFSMSrmm.

Do you want to test DFSMSrmm functions? If you are using another tape management system and are thinking about converting to DFSMSrmm, you can start DFSMSrmm and run it in parallel with your current system for testing purposes.

This book is intended to be a starting point for new professionals and a handbook for using the basic DFSMSrmm functions.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-5983-04 ISBN 0738439568