z/OS

DFSMShsm Implementation and Customization Guide

Version 2 Release 2

SC23-6869-02

Note

Before using this information and the product it supports, read the information in "Notices" on page 399.

This edition applies to Version 2 Release 2 of z/OS (5650-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 6. DFSMShsm starter set. . . . 101

Chapter 7. DFSMShsm sample tools 151

Chapter 8. Functional verification procedure 153

Chapter 9. Authorizing and protecting DFSMShsm commands and resources 169

Figures

Tables

About this document

This document helps you implement and customize IBM® z/OS® DFSMShsm. Primarily, this document provides descriptions of DFSMShsm data sets and describes how to create DFSMShsm data sets, procedures, and parameter library members. Global resource serialization, the functional verification procedure, and the DFSMShsm starter sets are also explained. Finally, this document includes special considerations to consider before installing DFSMShsm and describes tuning patches supported by DFSMShsm.

For information about the accessibility features of z/OS, for users who have a physical disability, see Appendix B, “Accessibility,” on page 395.

Who should read this document

This document is intended for system programmers and system administrators responsible for implementing a unique DFSMShsm environment. Implementation is the activity of customizing one or more PARMLIB members (ARCSTRxx and ARCCMDxx), startup parameters, and the SETSYS, DEFINE, and ADDVOL commands that define a DFSMShsm environment in one or more address spaces in one or more z/OS images.

Major divisions of this document

This document is divided into two parts: one for implementation and the other for customization.

The following topics are found in **Part 1, “Implementing DFSMShsm,” on page 1**:

- **Chapter 1, “Introduction,” on page 3** introduces the reader to the scope of the task of implementing DFSMShsm. It describes the various methods you can use to implement DFSMShsm at your site.
- **Chapter 2, “Installation verification procedure,” on page 7** describes where to find the installation verification procedure used to test the SMP/E installation of DFSMShsm modules into the appropriate z/OS system libraries.
- **Chapter 3, “DFSMShsm data sets,” on page 9** describes DFSMShsm data sets, how to create them, how to allocate them, how to calculate their size, and considerations for their placement.
- **Chapter 4, “User data sets,” on page 59** describes the types of user data sets that are supported by data mover DFSMShsm and data mover DFSMSdss.
- **Chapter 5, “Specifying commands that define your DFSMShsm environment,” on page 67** describes the SETSYS commands that define the default processing environment for DFSMShsm.
- **Chapter 6, “DFSMShsm starter set,” on page 101** describes the starter set for the SMS environment.
- **Chapter 7, “DFSMShsm sample tools,” on page 151** identifies partitioned data sets (in particular, HSM.SAMPLE.TOOL) available after installation. Their members provide sample tools and jobs to help you perform specific tasks.
- **Chapter 8, “Functional verification procedure,” on page 153** describes the functional verification procedure (FVP) for DFSMShsm.

- **Chapter 9, "Authorizing and protecting DFSMShsm commands and resources," on page 169** describes the steps you must take to protect DFSMShsm resources with RACF®, a component of the Security Server for z/OS. It also describes how to authorize users to work with RACF profiles and FACILITY classes.
- **Chapter 10, "Implementing DFSMShsm tape environments," on page 189** describes how to use the SETSYS commands to implement a management policy for tapes, devices, and performance in an SMS or non-SMS environment.
- **Chapter 11, "DFSMShsm in a multiple-image environment," on page 253** describes information about using DFSMShsm in a multiple DFSMShsm host environment and describes how to define the primary processing unit. Also covered is information about the various ways DFSMShsm serializes data sets, including global resource serialization, in a multiple DFSMShsm host environment.

The following topics are found in **Part 2, "Customizing DFSMShsm," on page 277**:
- **Chapter 12, "DFSMShsm in a sysplex environment," on page 279** describes information about using DFSMShsm in a sysplex environment. It covers information about how to promote secondary hosts, how to use extended addressability for control data sets, and how DFSMShsm functions in a GRSplex.
- **Chapter 13, "Calculating DFSMShsm storage requirements," on page 293** provides information about customizing your computing system storage for DFSMShsm, including storage calculation work sheets.
- **Chapter 14, "DFSMShsm libraries and procedures," on page 297** describes how and where to create DFSMShsm procedures and parameter library members.
- **Chapter 15, "User application interfaces," on page 313** describes information about DFSMShsm application programs and how to invoke them.
- **Chapter 16, "Tuning DFSMShsm," on page 329** describes information that you can use to tune DFSMShsm through use of DFSMShsm-supported patches.
- **Chapter 17, "Special considerations," on page 373** describes information that you should consider before you install DFSMShsm.
- **Chapter 18, "Health Checker for DFSMShsm," on page 383** describes information about Health Checker for DFSMShsm.

Required product knowledge

You should be familiar with the basic concepts of DFSMS described in *z/OS DFSMS Introduction*. You are presumed to have a background in using TSO and an understanding of z/OS concepts and terms.

z/OS information

This information explains how z/OS references information in other documents and on the web.

When possible, this information uses cross document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see *z/OS Information Roadmap*.

To find the complete z/OS library, go to IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or provide any other feedback that you have.

Use one of the following methods to send your comments:

1. Send an email to mhvrcfs@us.ibm.com.
2. Send an email from the "Contact us" web page for z/OS (http://www.ibm.com/systems/z/os/zos/webqs.html).

Include the following information:

- Your name and address.
- Your email address.
- Your telephone or fax number.
- The publication title and order number:
  z/OS V2R2 DFSMShsm Implementation and Customization Guide
  SC23-6869-02
- The topic and page number that is related to your comment.
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

Do not use the feedback methods that are listed for sending comments. Instead, take one of the following actions:

- Contact your IBM service representative.
- Call IBM technical support.
- Visit the IBM Support Portal at z/OS Support Portal (http://www-947.ibm.com/systems/support/z/zos/).

Summary of changes

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

Summary of changes for z/OS Version 2 Release 2 (V2R2)

The following changes are made for z/OS Version 2 Release 2 (V2R2).

Changed

- Table 11 on page 174 has been updated for the UPDTCDS command.

New

- “Allowing DFSMShsm to create backup copies older than the latest retained-days copy” on page 370.
- “Defining the common dump queue environment” on page 98
- “Common dump queue configurations” on page 291
- “Enabling or disabling RC 20 through RC 40 ARCMDEXT return code for transitions” on page 370
- “Enabling FSR records to be recorded for errors, reported by message ARC0734I, found during SMS data set eligibility checking for primary space management” on page 371

Summary of changes for z/OS Version 2 Release 1 (V2R1) as updated September 2014

The following changes are made for z/OS V2R1 as updated September, 2014.

Changed

- “Setting on compress for dumps and ABARS” on page 337 has been updated for the ZCOMPRESS function.

Summary of changes for z/OS Version 2 Release 1

See the following publications for all enhancements to z/OS Version 2 Release 1 (V2R1):

- *z/OS Migration*
- *z/OS Planning for Installation*
- *z/OS Summary of Message and Interface Changes*
- *z/OS Introduction and Release Guide*

Part 1. Implementing DFSMShsm

The following information is provided in this topic:

- **Chapter 1, "Introduction," on page 3** introduces the reader to the scope of the task of implementing DFSMShsm. It describes the various methods you can use to implement DFSMShsm at your site.
- **Chapter 2, "Installation verification procedure," on page 7** describes the installation verification procedure used to test the SMP/E installation of DFSMShsm modules into the appropriate MVS™ system libraries.
- **Chapter 3, "DFSMShsm data sets," on page 9** describes DFSMShsm data sets, how to create them, how to allocate them, how to calculate their size, and considerations for their placement.
- **Chapter 4, "User data sets," on page 59** describes the types of user data sets that are supported by data mover DFSMShsm and data mover DFSMSdss.
- **Chapter 5, "Specifying commands that define your DFSMShsm environment," on page 67** describes the SETSYS commands that define the default processing environment for DFSMShsm.
- **Chapter 6, "DFSMShsm starter set," on page 101** describes the starter set for the SMS environment.
- **Chapter 7, "DFSMShsm sample tools," on page 151** identifies partitioned data sets (in particular, HSM.SAMPLE.TOOL) available after installation. Their members provide sample tools and jobs to help you perform specific tasks.
- **Chapter 8, "Functional verification procedure," on page 153** describes the functional verification procedure (FVP) for DFSMShsm.
- **Chapter 9, "Authorizing and protecting DFSMShsm commands and resources," on page 169** describes the steps you must take to protect DFSMShsm resources with RACF, a component of the Security Server for z/OS. It also describes how to authorize users to work with RACF profiles and FACILITY classes.
- **Chapter 10, "Implementing DFSMShsm tape environments," on page 189** describes how to use the SETSYS commands to implement a management policy for tapes, devices, and performance in an SMS or non-SMS environment.
- **Chapter 11, "DFSMShsm in a multiple-image environment," on page 253** describes information about using DFSMShsm in a multiple DFSMShsm host environment and describes how to define the primary processing unit. Also covered is information about the various ways DFSMShsm serializes data sets, including global resource serialization, in a multiple DFSMShsm host environment.

Chapter 1. Introduction

This information is intended to help you implement your site's unique DFSMShsm environment. Implementation is the activity of specifying (in one or more PARMLIB members ARCSTR*xx* and ARCCMD*xx*) the startup parameters and SETSYS, DEFINE, and ADDVOL commands that define your site's DFSMShsm environment in one or more address spaces in one or more z/OS images.

ARCCMD*xx* is the name of a member in a PARMLIB that contains the commands that define one or more DFSMShsm hosts. When an instance of DFSMShsm is started, it reads its corresponding ARCCMD*xx* member to obtain environmental information about the way you have defined DFSMShsm.

Throughout this text the ARCCMD*xx* member and the starter set are referred to for hints and examples of DFSMShsm implementation. Recommendations for optimum performance and productivity are made, and current hardware and software technologies are emphasized.

Implementation begins after the System Modification Program/Extended (SMP/E) installation of the DFSMS program modules and only after a successful pass of the DFSMShsm installation verification procedure (IVP).

How to implement DFSMShsm

To implement DFSMShsm, perform the following steps and subtasks:

1. Install the z/OS DFSMS program product files in the appropriate system libraries with System Modification Program/Extended (SMP/E).

   Guideline: Place the IGX00024 load module in the fixed link pack area (FLPA) during the initial program load (IPL).
2. Edit and run the DFSMShsm installation verification procedure (IVP). For more information about the IVP, see Chapter 2, "Installation verification procedure," on page 7.
3. Implement DFSMShsm by either:
   - Expanding the starter set. The starter set is a sample DFSMShsm environment and is described in Chapter 6, "DFSMShsm starter set," on page 101.
   - Creating a parameter library (PARMLIB) startup member. By specifying the SETSYS, DEFINE, and ADDVOL commands in an ARCCMD*xx* PARMLIB member, you define an DFSMShsm environment. For details about the startup procedure and the ARCCMD*xx* PARMLIB member, see Chapter 14, "DFSMShsm libraries and procedures," on page 297.
4. After you have created a usable version of DFSMShsm, either with the starter set or by creating your own ARCCMD*xx* PARMLIB member, validate the basic functions of DFSMShsm by running the functional verification procedure (FVP). For information about the FVP, see Chapter 8, "Functional verification procedure," on page 153.

Starter set

The DFSMShsm starter set provides an example DFSMShsm environment. The starter set is as generic as possible to accommodate most environments, but it can be expanded and tested gradually as you implement additional DFSMShsm function.

The starter set, found in member ARCSTRST, is shipped as part of the DFSMShsm product and is for anyone needing a quick way of starting DFSMShsm in a nonproduction test environment. The starter set can be updated later with additional ARCCMD*xx* members and adapted to a production environment. (See "Starter set adaptation jobs" for more information.)

For more information about the starter set, see Chapter 6, "DFSMShsm starter set," on page 101.

Starter set adaptation jobs

The following jobs give some examples of adapting DFSMShsm to your environment. With some initial planning, these jobs can start managing data in a production mode. For more information about planning the DFSMS environment, see *z/OS Migration*.

Member
    Description

ARCCMD90
    defines a primary volume and a migration level 1 volume to DFSMShsm.

ARCCMD01 and ARCCMD91
    define a migration level 2 tape volume to DFSMShsm.

HSMHELP
    provides help text about DFSMShsm-authorized commands for users with data base authority.

HSMLOG
    provides a sample job to print the DFSMShsm log.

HSMEDIT
    provides a sample job to print the edit-log.

ALLOCBK1
    provides a sample job to allocate four backup versions of each control data set.

ALLOCBK2
    provides a sample job to allocate one backup copy of each control data set.

ALLOSDSP
    provides an example of how to allocate a small-data-set-packing data set.

HSMPRESS
    reorganizes the control data sets.

The starter set adaptation jobs, found in member ARCSTRST, are for anyone adapting DFSMShsm to a production environment. The jobs can be run after running the starter set, the FVP, or both.

Functional verification procedure (FVP)

The functional verification procedure (FVP) tests and verifies major DFSMShsm functions such as:

- Backup
- Dump
- Migrate
- Recall
- Recovery
- Restore

The FVP creates test data sets and exercises fundamental DFSMShsm processing. The FVP, found in member ARCFVPST, is for anyone wanting a thorough test of DFSMShsm using actual data. The FVP can be run after running the starter set but before running any of the jobs that adapt the starter set to your environment.

For more information on the functional verification procedure, see Chapter 8, "Functional verification procedure," on page 153.

Chapter 2. Installation verification procedure

The DFSMShsm installation verification procedure (IVP) is an optional procedure that verifies that DFSMShsm is correctly installed and can be started and stopped using a minimum of DASD resources. For more information on performing the IVP, see *z/OS V2R2 Program Directory* .

Chapter 3. DFSMShsm data sets

The DFSMShsm program requires the following data sets to support full-function processing:
- Migration control data set (MCDS)
- Backup control data set (BCDS)
- Offline control data set (OCDS)
- Journal data set
- Control data set and journal data set backup copies
- Problem determination aid (PDA) log data sets
- DFSMShsm logs
- Activity log data sets
- Small-data-set-packing (SDSP) data sets
- System data sets

For each of these data sets, we offer a description, tips on implementation, and a storage-size calculation work sheet to help you implement these data sets as you install DFSMShsm. Each data set discussion includes pertinent information such as maximum record size, data set type, storage guidance, and whether the data set is allocated by the starter set.

DFSMShsm supports VSAM KSDS extended addressability (EA) capability that uses any of the following for its control data sets: record level sharing (RLS) access mode, CDSQ serialization, or CDSR serialization.

Examples and discussions are based on 3390 DASD and a single DFSMShsm-host environment.

For more information about ...	See ...
DFSMShsm starter set	Chapter 6, "DFSMShsm starter set," on page 101
VSAM extended addressability capabilities	"Using VSAM extended addressability capabilities" on page 40
VSAM record level sharing	"Using VSAM record level sharing" on page 32
Multiple DFSMShsm-host environments	Chapter 11, "DFSMShsm in a multiple-image environment," on page 253

Control and journal data sets

The DFSMShsm control data sets are the resources with which DFSMShsm manages the storage environment. The journal data set receives an entry for each critical update to any CDS. Discussions of the control data sets and the journal are not complete without also discussing the CDS and journal backup copies. By retaining backup copies of the control data sets and the journal, you provide data integrity because you can merge the journal with the control data sets to reconstruct your environment to the point of failure should the control data sets be lost or damaged.

The three DFSMShsm virtual storage access method (VSAM) control data sets are the migration control data set (MCDS), the backup control data set (BCDS), and the offline control data set (OCDS). The MCDS manages migrated data sets, the BCDS manages backup versions and dump copies, and the OCDS manages backup and migration tape volumes. A single journal data set records each update to any of the three control data sets.

Preventing interlock of control data sets

In a multiple DFSMShsm-host environment, you must provide protection to prevent control data sets from entering into interlock situations. If you do not have a Global Resource Serialization-like product installed, you must protect your control data sets by ensuring that the following situations/conditions occur:

- That a CDS and the catalog where it is located are on the same volume. If you allocate a CDS on a volume other than the volume containing the catalog, interlock problems can occur.
- That control data sets are not placed on migration level 1 volumes. Placing control data sets on migration level 1 volumes may lead to degraded performance or a lockout in a multiple DFSMShsm-host environment.

  Volumes containing control data sets are subject to reserves in a multiple DFSMShsm-host environment. Migration level 1 volumes are also subject to reserves when small-data-set-packing data sets are accessed.
- That each CDS has a unique high-level qualifier that is cataloged in the user catalog on its respective volume if all control data sets in the catalog cannot reside on the same volume.
- That control data sets are not placed on the same volume with system resource data sets.

Migration control data set

The migration control data set (MCDS) provides information about migrated data sets and the volumes they migrate to and from. Additionally, internal processing statistics and processing-unit-environment data reside in the MCDS. DFSMShsm needs this information to manage its data sets and volumes. With this information you can verify, monitor, and tune your storage environment.

Migration Control Data Set (MCDS)

Required:
Yes.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Maximum record size:
2040 bytes.

Data set type:
VSAM key-sequenced data set (KSDS).

Storage guidance:
See "Considerations for DFSMShsm control data sets and the journal" on page 29.

An MCDS residing on a DASD device that is shared by two or more processors defines a multiple DFSMShsm-host environment to DFSMShsm. DFSMShsm assumes that there is more than one host sharing the MCDS, BCDS (optional), and OCDS (optional), if the MCDS is on shared DASD or the user specifies CDSSHR=RLS or CDSSHR=YES.

The control data sets are backed up at the beginning of automatic backup and are backed up differently from user data sets. The way control data sets are backed up depends on how you specify the SETSYS CDSVERSIONBACKUP command. For more information about specifying the CDS backup environment, see "Defining the backup environment for control data sets" on page 22.

For information about control data sets in a multiple DFSMShsm-host environment, see "CDS considerations in a multiple DFSMShsm host environment" on page 269.

DFSMShsm also writes two kinds of statistics records in the MCDS: daily statistics records and volume statistics records. DFSMShsm updates these records every hour if any data sets have changed. DFSMShsm writes the two statistics records to the system management facilities (SMF) data sets if you specify the SETSYS SMF command.

Migration control data set size

Figure 1 on page 12 is an example, based on 3390 DASD, of the MCDS size work sheet (also found in Appendix A, "DFSMShsm work sheets," on page 387). An example calculation follows the work sheet.

To calculate the size of the MCDS, you need to know only the number of data sets that you want to migrate.

Note:

1. VSAM extended addressability capabilities allow each MCDS cluster to exceed the 4 GB size. In addition, the MCDS can span up to four unique KSDS clusters. For more information about using VSAM extended addressability capabilities, see "Using VSAM extended addressability capabilities" on page 40. For more information about using KSDS clusters, see "Using multicluster control data sets" on page 34.

2. If Fast Subsequent Migration is activated via SETSYS TAPEMIGRATION(RECONNECT(ALL)) or SETSYS TAPEMIGRATION(RECONNECT(ML2DIRECTEDONLY)), MCD records are kept longer in the MCDS.

MCDS size work sheet: Use the following work sheet to calculate the size for your MCDS.

Migration Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = *mds* - Number of data sets that you want to migrate.

2. Substitute the value for *mds* in the following calculation. This is the space for your current MCDS records.

   516 x (*mds*= ________) = *subtotal* = ________________

3. Multiply the *subtotal* by 1.5 to allow for additional MCDS growth.

   subtotal x 1.5 = *total* = ________________

Total = **total number of bytes for the MCDS**

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the MCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the MCDS.

   Total bytes used by the MCDS / Total bytes per cylinder (3390) = (Total = ________) / 737280 = ________________

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the migration control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C156

Figure 1. Migration Control Data Set Size Work Sheet. This work sheet is also found in Appendix A, “DFSMShsm work sheets,” on page 387.

Example of a MCDS calculation: The following example uses the work sheet in Figure 1 and the following assumptions to calculate how many cylinders are needed for the MCDS.

1. Assuming that your environment consists of the following:

   `150 000 data sets that you want to have migrated (`*mds*`)`

2. Substitute the value for *mds* into the following calculation. This is the space for your current MCDS records.

   `516 x (`*mds*` = 150 000) = `*subtotal*` = 77 400 000`

3. Multiply the *subtotal* by 1.5 to allow for additional MCDS growth.

   `77 400 000 x 1.5 = 116 100 000`

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes for the MCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the MCDS.

$$\frac{\text{Total bytes used by the MCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{(\textit{total} = 116\,100\,000)}{737\,280} = \text{157.4 or approximately 158 cylinders}$$

Estimating the number of data sets (mds) using DCOLLECT: This IDCAMS DCOLLECT procedure is designed for customers with only active data (having no migrated data). Use DCOLLECT to obtain information (D type records) on the number of active data sets in your storage environment.

You can use the number of D type records that are written to the "OUTDS" data set to estimate the number of active data sets. In the previous calculation, substitute the number of D type records for *mds*. For more information on IDCAMS DCOLLECT command and all of its parameters, refer to *z/OS DFSMS Access Method Services Commands*.

```
//COLLECT1 JOB    ...
  //STEP1    EXEC   PGM=IDCAMS
  //SYSPRINT DD     SYSOUT=A
  //OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
  //       STORCLAS=LARGE,
  //       DSORG=PS,
  //       DCB=(RECFM=VB,LRECL=644,BLKSIZE=0),
  //       SPACE=(1,(100,100)),AVGREC=K,
  //       DISP=(NEW,CATLG,KEEP)
  //SYSIN    DD     *
       DCOLLECT -
             OFILE(OUTDS) -
   NOVOLUMEINFO
  /*
```

Backup control data set

The backup control data set (BCDS) provides DFSMShsm with information defining the backup and dump environment. DFSMShsm needs this information to manage its backup data sets and volumes. DFSMShsm updates the contents of the BCDS with current backup version information.

Backup Control Data Set (BCDS)

Required:
No, not for space management and general DFSMShsm processing. DFSMShsm requires a BCDS if you want to use the incremental backup, dump, ABARS, or fast replication functions.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Maximum record size:

- 6544 bytes, if you need 100 or less *active* backup versions of any data set, if you use dump class stacking values greater than 99, or if you use fast replication.
- 2093 bytes, if you need 29 or less *active* backup versions of any data set, or if you use dump class stacking values of 99 or less.
- 2040 bytes, if you need 29 or less *active* backup versions of any data set, or if you use dump class stacking values of 97 or less.

Note:

1. The total number of backup versions for a data set may exceed the maximum number of *active* backup versions when a RETAINDAYS value is specified. Additional records are created to contain the retained backup versions. For more information about specifying a RETAINDAYS value, see the BACKDS command in *z/OS DFSMShsm Storage Administration*.
2. Any time the BCDS maximum record size is changed, the BCDS must be reorganized.

Data set type:
VSAM key-sequenced data set (KSDS).

Storage guidance:
See "Considerations for DFSMShsm control data sets and the journal" on page 29.

The control data sets are backed up at the beginning of automatic backup and are backed up differently from user data sets. The way control data sets are backed up depends on how you specify the SETSYS CDSVERSIONBACKUP command. For more information about specifying the CDS backup environment, see "Defining the backup environment for control data sets" on page 22.

For information about the control data sets in a multiple DFSMShsm-host environment, see "CDS considerations in a multiple DFSMShsm host environment" on page 269.

Backup control data set size

The BCDS must be large enough to contain the maximum number of records for all the data sets and volumes that have been backed up.

Figure 2 on page 15 is an example, based on 3390 DASD, of the BCDS size work sheet (also found in Appendix A, "DFSMShsm work sheets," on page 387). An example calculation follows the work sheet.

To calculate the size of your backup control data set, you need to know the number of backup versions you want to keep and the number of data sets that you want to have automatically backed up.

Note: VSAM extended addressability capabilities allow each BCDS cluster to exceed the 4 GB size. In addition, the BCDS can span up to four unique KSDS clusters. For more information about using VSAM extended addressability capabilities, see "Using VSAM extended addressability capabilities" on page 40. For more information about using KSDS clusters, see "Using multicluster control data sets" on page 34.

BCDS size work sheet: Use the following work sheet to calculate the size for your BCDS.

Backup Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = *bver* - Number of backup versions of each data set. This same number is used with the VERSION parameter of the SETSYS command or is specified in management classes. The upper bound for *bver* is either 29 or 100, depending on the maximum record size in the BCDS definition.

   ________ = *nds* - Number of data sets backed up automatically.

2. Substitute the values for *bver* and *nds* in the following calculation. This is the space for your current BCDS records.

   398 x *(bver=* ________ *)* x *(nds=* ________ *) = subtotal* = ______________

3. Multiply the *subtotal* by 1.5 to allow for additional BCDS growth.

   subtotal x 1.5 = *total* = ______________

Total = **total number of bytes for the BCDS**

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the BCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the BCDS.

   $$\frac{\text{Total bytes used by the BCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{\text{(Total = \_\_\_\_\_\_\_\_)}}{737280} = \text{\_\_\_\_\_\_\_\_\_\_\_\_}$$

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the backup control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C155

Figure 2. Backup Control Data Set Size Work Sheet. This work sheet is also found in Appendix A, "DFSMShsm work sheets," on page 387.

Example of a BCDS calculation: This example uses the work sheet in Figure 2 and several assumptions about the environment to calculate how many cylinders are needed for the BCDS.

1. Assuming that your environment consists of the following:

5	average number of backup versions (active and retained) of any data set (*bver*)
150 000	data sets that you want automatically backed up (*nds*)

   calculate the space for your current BCDS records, using this formula:

   ```
   398  x  bver  x  nds   =  subtotal
   ```

Substituting the values for *bver* and *nds* results in:

```
398  x  5  x  150 000   =  298 500 000
```

2. Multiply the subtotal by 1.5 to allow for additional BCDS growth.

```
298 500 000 x  1.5  =  447 750 000
```

3. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the BCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the BCDS.

$$\frac{\text{Total bytes used by the BCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{447\ 750\ 000}{737\ 280} = \text{607.23 or approximately 607 cylinders}$$

Estimating the number of data sets (bver) using DCOLLECT: This IDCAMS DCOLLECT procedure is designed for customers with only active data (having no migrated data). Use DCOLLECT to obtain information (D type records) on the number of active data sets in your storage environment.

You can use the number of D type records that are written to the "OUTDS" data set to estimate the number of active data sets. In the previous calculation, substitute the number of D type records for *bver*. For more information on IDCAMS DCOLLECT command and all of its parameters, refer to *z/OS DFSMS Access Method Services Commands*.

```
//COLLECT1 JOB    ...
  //STEP1    EXEC   PGM=IDCAMS
  //SYSPRINT DD     SYSOUT=A
  //OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
  //       STORCLAS=LARGE,
  //       DSORG=PS,
  //       DCB=(RECFM=VB,LRECL=644,BLKSIZE=0),
  //       SPACE=(1,(100,100)),AVGREC=K,
  //       DISP=(NEW,CATLG,KEEP)
  //SYSIN    DD     *
       DCOLLECT -
             OFILE(OUTDS) -
   NOVOLUMEINFO
  /*
```

Offline control data set

The offline control data set (OCDS) provides DFSMShsm with information about each migration and backup tape and about each data set residing on these tapes.

Offline Control Data Set (OCDS)

Required:
No, but strongly recommended. The offline control data set is required if you are using tapes in your environment.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Maximum record size:
- 6144 bytes, if you plan to you plan to use extended tape table of contents (TTOCs) at your installation. This record size allows up to 106 data set entries per record.
- 2040 bytes, if you need no more than 33 data set entries per record.

Requirement: The maximum record size must be 6144 bytes if you plan to use extended TTOCs. Using extended TTOCs also requires that you issue the SETSYS EXTENDEDTTOC(Y) command on each host that will share the OCDS. For more information on the SETSYS command, see *z/OS DFSMShsm Storage Administration*.

Data set type:
VSAM key-sequenced data set (KSDS).

Storage guidance:
See "Considerations for DFSMShsm control data sets and the journal" on page 29.

TTOC records are written to the OCDS. Tape Copy Needed (TCN) records are written to the OCDS if an exception occurs during duplex processing.

Control data sets are backed up at the beginning of automatic backup and are backed up differently than user data sets, depending on how you specify the SETSYS CDSVERSIONBACKUP command. For more information, see "Defining the backup environment for control data sets" on page 22.

For information about the control data sets in a multiple DFSMShsm-host environment, see "CDS considerations in a multiple DFSMShsm host environment" on page 269.

Offline control data set size

The OCDS must be large enough to contain the TTOC records for each migration and backup tape volume managed by DFSMShsm. You can define the OCDS with a maximum record size of 6144 bytes, which allows up to 106 data sets entries per tape table of contents record entry.

Figure 3 on page 18 is an example, based on 3390 DASD, of the offline control data set work sheet (also found in Appendix A, "DFSMShsm work sheets," on page 387). The work sheet helps you determine the optimum storage size for the OCDS. Insert values in the legend variables for your installation, add these values, and get a close approximation of the size for the OCDS.

VSAM extended addressability capabilities allow the OCDS cluster to exceed the 4 GB size. The OCDS is limited to 1 cluster. For more information about using VSAM extended addressability capabilities, see "Using VSAM extended addressability capabilities" on page 40.

OCDS size work sheet: To calculate the size for your OCDS, use the work sheet in Figure 3 on page 18. This work sheet assumes a larger record size (6144 bytes) to enable 106 data set entries for extended TTOCs.

Offline Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

________	= bver	- Number of backup data set versions that the volume contains.
________	= mds	- Number of migration data set copies the volume contains.
________	= nds	- Number of data sets backed up automatically.
________	= n	- Total number of backup version and migration copy data sets for your installation.

2. Substitute the value for *n* in the following calculation. This is the space for your current OCDS records.

$$\frac{n}{106} \times 6144 = \text{subtotal} \qquad = \_\_\_\_\_\_\_\_\_\_$$

3. Multiply the *subtotal* by 1.5 to allow for additional OCDS growth.

subtotal x 1.5 = *total* = ________

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the OCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the OCDS.

$$\frac{\text{Total bytes used by the OCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{(\text{Total} = \_\_\_\_\_\_)}{737280} = \_\_\_\_\_\_\_\_\_\_$$

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the offline control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C157

Figure 3. Offline Control Data Set Size Work Sheet. This work sheet is also found in Appendix A, "DFSMShsm work sheets," on page 387.

Example of an OCDS calculation: In this example, the work sheet in Figure 3 and the following assumptions are used to calculate how many cylinders are needed for the OCDS.

1. Assuming that you are using extended TTOCs and that **200 000 mds + (nds** × **bver)** is the total number of backup data set versions and migration data set copies on tape at your site.

```
200 000
_______   x   6144  =  subtotal         = 11 592 452
  106
```

2. Substitute the preceding value in the following calculation. This is the space for your current OCDS records.
3. Multiply *subtotal* by 1.5 to allow for additional OCDS growth.

```
11 592 452   x   1.5   =  17 388 678
```

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes used by the OCDS. If the result is a fraction, round up to the next whole number. This number is the number of cylinders you should allocate for the OCDS data set.

Total bytes used by the OCDS	=	(*total* = 17 388 678)	=	23.6 or approximately 24 cylinders
Total bytes per cylinder (3390)		737 280		

If your installation has only active data (that is, no migrated data), see "Estimating the number of data sets (mds) using DCOLLECT" on page 13 and "Estimating the number of data sets (bver) using DCOLLECT" on page 16 for information about how to estimate the number of active data sets in your storage environment.

In the previous calculation, substitute the number of D type records for *mds* and *bver*.

Enabling the use of extended TTOCs

If you decide to change the size of a previously allocated and used OCDS to enable the use of extended TTOCs, use the procedure described in this topic.

Attention: In an HSMplex environment, you should not enable extended TTOCs on any host in the HSMplex until the shared OCDS has been redefined with a record size of 6144 bytes.

Perform these steps to enable the use of extended TTOCs.

1. Back-up your existing OCDS.
2. Define a new OCDS with a maximum record size of 6144 bytes.
3. Shutdown DFSMShsm.
4. Use the REPRO command to copy the old OCDS into the newly defined OCDS.
5. Ensure that the DFSMShsm startup procedure OFFCAT DD statement points to the new OCDS name.
6. Mark all backup and migration tapes that are partials as full by issuing DELVOL volser BACKUP|MIGRATION MARKFULL before restarting DFSMShsm.

   If you do not mark all partial tapes full, when DFSMShsm processing selects volumes, these partial tapes will be selected, the message ARC0309I TAPE VOLUME volser REJECTED, TTOC TYPE CONFLICT will be issued, and the tape will be marked full. DFSMShsm will process all partials, and mark them full before selecting a new scratch tape.
7. Restart DFSMShsm.
8. Issue the SETSYS EXTENDEDTTOC(Y) command.

The SETSYS EXTENDEDTTOC(Y) setting remains in effect for the duration of the DFSMShsm startup. To automatically enable extended TTOCs at DFSMShsm startup, add SETSYS EXTENDEDTTOC(Y) to the ARCCMD*xx* parmlib member in SYS1.PARMLIB.

Journal data set

The journal data set provides DFSMShsm with a record of each critical change made to a control data set from any host since the last time that CDS was successfully backed up. DFSMShsm recovers control data sets by merging the journal with a backed up version of the control data set. Control data sets cannot be recovered to the point of failure without the journal. Use of the journal is highly recommended.

The journal data set is a special type of data set. It contains a control record at the beginning that points to the last record written to the journal. However, the rest of the data set is processed as a sequential data set, containing a record of each significant change made to the MCDS, BCDS, and OCDS. The journal data set normally does not contain an end-of-file marker. Ensure that you allocate the journal data set only on a DASD volume.

In a multiple DFSMShsm-host environment, if hosts share a single set of control data sets, they must also share a single journal. All DFSMShsm recovery procedures are based on a single journal to merge with a backed up version of a control data set.

Journal Data Set

Required:
No, but strongly recommended.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical sequential data set.

Storage guidance:
The DFSMShsm journal must reside on a DASD device.

The first record of the journal keeps track of the location of the last record in the journal. This pointer is the full disk address. Therefore, the journal data set must not be moved to another location on the current volume or another volume.

To avoid a performance problem, the DFSMShsm journal must not share a volume with any system resource data sets or with any of the DFSMShsm control data sets.

The journal can be defined as a large format data set, which can exceed 65 535 tracks. Doing so can help to reduce the frequency of journal full conditions. For information about large format data sets, see *z/OS DFSMS Using Data Sets*.

The journal should not share an HDA with any of the CDS data sets. This is important, so that if physical damage occurs, both the journal and a CDS will not be lost at the same time.

The journal is backed up at the beginning of automatic backup (along with the control data sets) and is backed up differently from user data sets. The Journal and Control data sets must be backed up. For information about specifying the journal backup environment, see "Defining the backup environment for control data sets" on page 22.

Also see "Considerations for DFSMShsm control data sets and the journal" on page 29.

Journal data set size

The amount of space that you allocate for your journal data set depends on how often DFSMShsm automatically backs up the control data sets and the amount of DFSMShsm activity between backups. The journal data set requires sufficient storage to accumulate journal entries that occur between one control data set backup and the next. The automatic backup of the control data sets clears the contents of the journal.

Allocate the journal as a single volume, single extent data set that is contiguous and non-striped.

If you decide to change the size of a previously allocated and used journal, make sure that the newly allocated journal does not contain the EOF of the previous journal. You can do this in one of three ways:

- Rename the old journal, allocate the new journal, and then delete the old journal.
- Use IEBGENER (not IEFBR14) to allocate the new journal.
- Allocate your new journal on a different DASD volume.

Migrating the journal to a large format data set

Consider using a large format data set for the DFSMShsm journal. A large format data set is a physical data sequential data set with the ability to grow beyond 65 535 tracks per volume. Using a larger journal data set can allow more DFSMShsm activity to take place between journal backups and helps to avoid journal full conditions. For more information about large format data sets, see *z/OS DFSMS Using Data Sets*.

If you decide to migrate your current journal to a large format data set, use the following procedure:

1. Stop all but one of the DFSMShsm hosts.
2. Either enter the HOLD ALL command, or set DFSMShsm to emergency mode.
3. Back up the control data sets through the BACKVOL CDS command. Doing so creates a backup of the journal and nulls the journal.
4. Stop the remaining hosts.
5. Rename the current journal data set. Do not delete the data set yet because the new journal should not be allocated in place of the existing journal.
6. Allocate the new large format sequential journal data set. If you allocate the new journal data set with a name other than the old journal data set name, edit the DFSMShsm startup procedure to use the new journal data set name.
7. Restart the DFSMShsm hosts.

Because of the size of the journal, it might be impractical to perform Steps 5 and 6. As an alternative to these steps, you can delete the existing journal, allocate the new journal, and then run an IEBGENER or IDCAMS (using the REPRO command) job to copy zero records into the journal.

In the event that you allocate and use a large format sequential journal data set, and you need to convert back to a non-large format journal data set, use a slight variation of the preceding procedure to allocate a non-large format data set:

- In Step 5, rename the large format journal data set. Do not delete the data set yet because the new journal should not be allocated in place of the existing journal.
- In Step 6, allocate the new non-large format sequential journal data set.

Updating IFGPSEDI for the enhanced data integrity function

If you plan to activate the enhanced data integrity function (EDI), you must include the DFSMShsm journal in the IFGPSEDI member of parmlib.

Specifying the names of the backup data sets

Use subparameters of the SETSYS CDSVERSIONBACKUP command to specify the data set names that are to be assigned to the backup data sets. You can use up to 31 characters in the data set name, including periods, but you must not end the name with a period. DFSMShsm appends a qualifier to the data set name that you specify. The form of this qualifier depends on the data mover used to perform the backup and whether the CDS is a single or multicluster:

- DSy.V*nnnnnnn*-V specifies that DFSMShsm (via IDCAMS) was the data mover
- DSy.D*nnnnnnn*-D specifies that DFSMSdss was the data mover
- DSy.X*nnnnnnn*-X specifies that the backup failed, regardless of the data mover

Note: "*nnnnnnn*" is the version number. "y" is the cluster number for a multicluster CDS. ".DSy" does not exist for a single-cluster CDS.

If you do not specify the data set names, DFSMShsm generates names of *uid*.BCDS.BACKUP, *uid*.MCDS.BACKUP, *uid*.OCDS.BACKUP, and *uid*.JRNL.BACKUP for the data sets. DFSMShsm substitutes the UID from the startup procedure for *uid*.

The complete command that is added to the ARCCMDxx member is:

```
SETSYS CDSVERSIONBACKUP(DATAMOVER(DSS) -
   BACKUPCOPIES(4) -
   BACKUPDEVICECATEGORY(DASD) -
   MCDSBACKUPDSN(BHSM.MCDS.BACKUP) -
   BCDSBACKUPDSN(BHSM.BCDS.BACKUP) -
   OCDSBACKUPDSN(BHSM.OCDS.BACKUP) -
   JRNLBACKUPDSN(BHSM.JRNL.BACKUP))
```

Defining the backup environment for control data sets

When DFSMShsm is started, it gets environmental and function information from the parameter library (PARMLIB). The SETSYS, DEFINE, and ADDVOL commands that define the way in which DFSMShsm manages your data are in the PARMLIB member. (See "Starter set example" on page 109 in the starter set for an example of the PARMLIB member ARCCMD00.)

In the following text, how to define the CDS and journal backup environment is discussed in a step-by-step fashion. The SETSYS CDSVERSIONBACKUP command determines how DFSMShsm backs up your control data sets. Subparameters of the CDSVERSIONBACKUP command allow you to specify:

- The data mover that backs up the control data sets (DFSMSdss is recommended)
- The number of backup versions to keep for the control data sets
- The device type on which to store the backup versions of the control data sets
- The names of the backup version data sets

SMS storage groups and management classes allow you to control other functions:

- Prevent backup (outside of CDSVERSIONBACKUP) of the control data sets and the journal
- Prevent migration of the control data sets and the journal
- Specify that concurrent copy be used to backup the control data sets, assuming that the control data sets are on volumes connected to a controller that provides concurrent copy

Steps for defining the CDS and journal backup environment

The following is a sequence of steps for defining a CDS backup environment:

1. Add the SETSYS CDSVERSIONBACKUP command to the ARCCMD*xx* PARMLIB member.
2. Prevent the control data sets and the journal from being backed up as part of user data set backup. Control data sets and the journal are backed up separately as specified by the SETSYS CDSVERSIONBACKUP command.

   If the control data sets and journal are SMS-managed:
   - Place them on volumes that are defined in a storage group with

     AUTO BACKUP ===> NO

     or
   - Associate them with a management class whose attributes are

     AUTO BACKUP ===> N

   If the control data sets and the journal are non-SMS-managed, issue the ALTERDS command to prevent them from being backed up outside of CDSVERSIONBACKUP.
3. Prevent the control data sets and journal from migrating. Allowing the control data sets and the journal to migrate is inadvisable because you might not be able to recover should any of the control data sets be damaged.

   If the control data sets and journal are SMS-managed:
   - Place them on volumes that are defined in a storage group with

     AUTO MIGRATE ===> NO

     or
   - Associate them with a management class whose attributes are COMMAND OR AUTO MIGRATE ===> NONE

   If the control data sets and journal are non-SMS-managed, issue the SETMIG command to prevent them from migrating.
4. Determine whether your control data sets are backed up using concurrent copy. If you want your control data sets to be backed up using concurrent copy:
   - Ensure that they are associated with a management class BACKUP COPY TECHNIQUE attribute of REQUIRED(R), PREFERRED (P), VIRTUALREQUIRED (VR), VIRTUALPREFERRED (VP), CACHEREQUIRED (CR), or CACHEPREFERRED (CP).
   - Ensure that they are on a DASD volume with a concurrent-copy capable 3990 controller.
   - Ensure that you specify DATAMOVER(DSS).
5. Determine whether the data mover for the control data sets is DFSMShsm or DFSMSdss. DFSMSdss is recommended as the data mover, because DFSMSdss validates the control data sets during BACKUP and supports concurrent copy.

If you specify . . .	Then . . .
SETSYS CDSVERSIONBACKUP (DATAMOVER(DSS))	DFSMShsm invokes DFSMSdss to perform a logical dump of the control data sets and uses sequential I/O to back up the journal. DFSMSdss validates the control data sets while backing them up and uses concurrent copy if it was specified in the management class.

If you specify . . .	Then . . .
SETSYS CDSVERSIONBACKUP (DATAMOVER(HSM))	DFSMShsm exports the control data sets and backs up the journal with sequential I/0. The control data sets are not validated during backup.

6. Choose the number of backup versions you want to keep for the control data sets. The number of backup versions that DFSMShsm keeps is determined by the number you specify on the BACKUPCOPIES subparameter of the SETSYS CDSVERSIONBACKUP command.

   Note: Whenever DFSMShsm actively accesses the control data sets in RLS mode, DFSMSdss must be specified as the datamover for the CDS backup. If data is directed to tape, the PARALLEL parameter must also be specified. If either condition is not met during auto CDS version backup, these values override existing values and message ARC0793I is issued. If either of these conditions is not met when BACKVOL CDS is issued, the command fails.
7. Choose the device category (DASD or tape) on which you want DFSMShsm to back up your control data sets and journal. Parallel is faster than serial and is required in order to use concurrent copy.

If you specify . . .	Then . . .
SETSYS CDSVERSIONBACKUP (BACKUPDEVICECATEGORY (DASD))	DFSMShsm always backs up the control data sets in parallel to DASD devices.

If you are backing up the control data sets and the journal to DASD, you must preallocate the backup version data sets. You can preallocate the DFSMShsm CDS and journal data set by running the starter set job "ALLOCBK1" on page 142 before starting DFSMShsm.

If you specify . . .	Then . . .
SETSYS CDSVERSIONBACKUP (BACKUPDEVICECATEGORY (TAPE))	DFSMShsm backs up the control data sets to tape.

Whether tape CDS backups are in parallel is determined by the data mover you specify and the optional PARALLEL|NOPARALLEL option for DFSMShsm control data set backup.

If you specify . . .	Then . . .
SETSYS CDSVERSIONBACKUP (DATAMOVER(DSS))	DFSMSdss backs up the control data sets to tape in parallel. Concurrent copy can be used.
SETSYS CDSVERSIONBACKUP (DATAMOVER(HSM))	DFSMShsm backs up the control data sets serially. Concurrent copy is not available, and the control data sets are not validated during backup.
SETSYS CDSVERSIONBACKUP (DATAMOVER(HSM) PARALLEL)	DFSMShsm backs up the control data sets to tape in parallel. Concurrent copy is not available and the control data sets are not validated during backup. For more information about the PARALLEL|NOPARALLEL tape option, refer to *z/OS DFSMShsm Storage Administration*.

If you are backing up the control data sets and the journal to tape, DFSMShsm dynamically allocates scratch tape volumes so you need not preallocate backup version data sets.

If you are backing up control data sets to DASD, you must catalog the CDS version backup data sets on all systems that are eligible for primary host promotion.

8. Determine the names for the backup data sets.

   You specify the names that are assigned to the backup version data sets when you specify the MCDSBACKUPDSN, BCDSBACKUPDSN, OCDSBACKUPDSN, and the JRNLBACKUPDSN subparameters of the SETSYS CDSVERSIONBACKUP command. The backup version data set names can be up to 35 characters (including periods) and cannot end in a period.

Figure 4 is an example of the SETSYS CDSVERSIONBACKUP command and its subparameters, as it would appear in PARMLIB member ARCCMD*xx*.

```
/***********************************************************************/
/* SAMPLE SETSYS CDSVERSIONBACKUP COMMAND AND SUBPARAMETERS THAT       */
/* DEFINE A CDS BACKUP ENVIRONMENT WHERE DSS BACKS UP FOUR COPIES OF   */
/* THE CDSS IN PARALLEL TO DASD.                                       */
/***********************************************************************/
/*
SETSYS CDSVERSIONBACKUP(DATAMOVER(DSS) -
   BACKUPCOPIES(4) -
   BACKUPDEVICECATEGORY(DASD) -
   MCDSBACKUPDSN(BHSM.MCDS.BACKUP) -
   BCDSBACKUPDSN(BHSM.BCDS.BACKUP) -
   OCDSBACKUPDSN(BHSM.OCDS.BACKUP) -
   JRNLBACKUPDSN(BHSM.JRNL.BACKUP))
/*
```

Figure 4. Example CDSVERSIONBACKUP Command

Improving performance of CDS and journal backup

System performance is directly related to the time it takes to back up the control data sets and the journal, because all DFSMShsm activity and all JES3 setup and initialization activity is suspended while these important data sets are backed up. By improving the performance of CDS and journal backup, you can decrease the time that system-wide serialization is in effect.

To decrease the time required to back up the control data sets and the journal, consider any or all of the following:

- Use the non-intrusive journal backup method. This method allows DFSMShsm activity to continue while the journal is backed up. For more information about the non-intrusive journal backup method, see the topic about using non-intrusive journal backup in *z/OS DFSMShsm Storage Administration*.
- Activate XCF and specify PLEXNAME via the SETSYS command for each of your multi-host DFSMShsm systems. For more information, see the SETSYS command in *z/OS DFSMShsm Storage Administration*.
- Back up the control data sets using concurrent copy by allocating them on SMS-managed, concurrent-copy-capable devices (does not apply to the journal) and specifying DATAMOVER(DSS). Ensure that they are associated with a management class BACKUP COPY TECHNIQUE attribute of REQUIRED(R), PREFERRED (P), VIRTUALREQUIRED (VR), VIRTUALPREFERRED (VP), CACHEREQUIRED (CR), or CACHEPREFERRED (CP).
- Modify the block size for CDS backup version data sets.

Preallocate DASD backup data set copies with a block size equal to one-half the track size of the DASD device. For example, the half-track capacity of a 3390 device is 27 998.

If the keyword BLKSIZE is specified on the pre-allocated DASD backup data set copies, it must be in the range of 7892 to 32 760 inclusive.

- Back up the control data sets in parallel.

  If you are backing up your control data sets to DASD (BACKUPDEVICECATEGORY(DASD)), the control data sets are always backed up in parallel.

  If you are backing up your control data sets to tape (SETSYS BACKUPDEVICECATEGORY(TAPE)), control data sets are always backed up in parallel when you:

  – Specify DATAMOVER(DSS)
  – Specify DATAMOVER(HSM) and specify the TAPE(PARALLEL) option of the CDSVERSIONBACKUP command. If you choose parallel backup to tape, ensure that one tape drive is available to backup each CDS cluster and another tape drive is available to backup the journal data set.

    Requirement: If a CDS is a multicluster CDS, then you need additional tape drives.

    For more information about the TAPE(PARALLEL|NOPARALLEL) option of the SETSYS CDSVERSIONBACKUP command, see the SETSYS command in *z/OS DFSMShsm Storage Administration*.

Table 1 shows the interaction of the BACKUPDEVICECATEGORY and DATAMOVER subparameters in defining the parallel backup environment for control data sets.

Table 1. Backing up the control data sets in Parallel

Subparameters of the SETSYS CDSVERSIONBACKUP command	RESULT
BACKUPDEVICECATEGORY (DASD) DATAMOVER(HSM)	Parallel backup, preallocate backup data sets
BACKUPDEVICECATEGORY (DASD) DATAMOVER(DSS)	Parallel backup, preallocate backup data sets
BACKUPDEVICECATEGORY (TAPE) DATAMOVER(HSM)	No parallel backup unless you specify TAPE(PARALLEL)
BACKUPDEVICECATEGORY (TAPE) DATAMOVER(DSS)	Parallel backup, allocate a tape drive for the journal and one for each CDS cluster

Monitoring the control and journal data sets

DFSMShsm monitors the space used by the three control data sets and the journal data set and informs the system operator with a warning message when the space used exceeds a specified threshold. This threshold can be different for each control data set and different for the journal data set. You can express the threshold as a percentage with the SETSYS MONITOR command, or you can accept the DFSMShsm defaults (threshold of 80%). When the space that is used by the data set exceeds this percentage of total space, DFSMShsm issues warning message ARC0909E or ARC0911E to the operator. If no records are written because the MCDS, BCDS, or OCDS is full, DFSMShsm holds the requested function and issues warning message ARC0910E to the operator. Depending on the message, the operator can schedule either an increase in journal size or a reorganization of the control data sets.

Space information is obtained for each control data set and for the journal data set. This information includes the total space allocated for each data set and the high used relative byte address (RBA).

DFSMShsm monitors information about space use and the threshold values for the control data sets and the journal data set. The system programmer can use the SETSYS MONITOR command to:

- Specify which informational messages DFSMShsm prints at the operator console for each data set
- Choose the threshold value of the specified total space that DFSMShsm uses when monitoring space, or use the default of 80%

Reorganizing the control data sets

Reorganization of the control data sets is extremely dangerous, and reorganization should be performed only if one of the following conditions exists:

- When you need to move a control data set to a new DASD volume.
- When you need to increase the space allocation of a control data set, as indicated by the ARC0909E or ARC0911E messages.

DFSMShsm issues these messages when the threshold requested by the SETSYS MONITOR command has been exceeded. You can display the current status of the control data sets with the QUERY CONTROLDATASETS command. Status is reported in the ARC0148I and ARC0948I messages. To reorganize the control data sets, use the Access Method Services (AMS) EXPORT and IMPORT commands. To move the control data sets to another volume or to increase their space allocations, use the AMS DEFINE, REPRO, and DELETE commands.

Attention: Do not attempt to reorganize your control data sets while DFSMShsm is running on any processor that uses those control data sets.

Several topics should be considered when reorganizing the control data sets. The primary consideration is that DFSMShsm must be shut down in all processors that have access to the control data sets. No job in any processor should attempt accessing the control data sets during their reorganization. The starter set job, “HSMPRESS” on page 147, provides a sample job for reorganizing control data sets. “VSAM SHAREOPTIONS parameters for control data sets” on page 270 provides detailed information about protecting the control data sets from damage when they are reorganized in a multiple DFSMShsm-host environment.

After a control data set is reorganized, expect to see performance degraded for a period of about three weeks while CA and CI splits occur. The 44-character keys for most of the DFSMShsm records are time-stamped to distribute the I/O activity as evenly as possible. When increasing the space allocation, attempt a large enough increase so that reorganization is not required for another 3–6 months, based on your current growth rate.

You can significantly reduce the need to reorganize the DFSMShsm control data sets by enabling the CA reclaim function for them. For more information, see the topic about reclaiming CA space in *z/OS DFSMS Using Data Sets*.

Control data set and journal data set backup copies

The control data set (CDS) and journal data set backup copies provide you with the ability to conduct on-site recovery should any of the control data sets become lost or damaged. The CDS and journal backup copies are optional but are highly recommended because they enable you to reconstruct your control data sets to the point of failure.

Control Data Set Backup Copies

Required:
No, but highly recommended.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical sequential.

Storage guidance:
Do not allow DFSMShsm to inadvertently migrate or backup your control data set and journal data set backup copies. If the CDS and journal backups are placed on DFSMShsm-managed volumes, ensure that policies are in place to prevent their migration or backup.

For information on the methods to back up the control data sets, see "Phase 1: Backing up the control data sets" in *z/OS DFSMShsm Storage Administration*.

Storage guidance for control data set and journal data set backup copies

Ensure that control data sets and journal data set backup copies are not backed up or migrated. Backing up CDS backup copies is unnecessary and inefficient. Migrating CDS backup copies causes problems because DFSMShsm is unable to access them or recover the control data sets. Consider, for a moment, that your backup copy of the MCDS has migrated and your current MCDS is damaged. The data needed to recall your backup copy is in the damaged (and unusable) current MCDS. If you have no other volume available to put the backup data sets on, and if the volume is SMS-managed, the backup copies must be associated with the NOMIG attribute which prevents them from being migrated. If the CDS and journal backup copies are non-SMS-managed, you can specify a SETMIG command for each of the backup versions of the control data sets and for the journal. For more information about the SETMIG command, see *z/OS DFSMShsm Storage Administration*. .

Size of the control data set and journal data set backup copies

The amount of space that you allocate for your CDS and journal backup copies depends on the size of the CDS or journal data set being backed up. Ensure that all DASD backup data sets have the same size as their corresponding control data sets or journal data set.

Example: For a backup data set of the MCDS, allocate a DASD backup data set that is the same size as the MCDS. For a backup data set of the BCDS, allocate a DASD backup data set that is the same size as the BCDS, and so forth.

If the journal is allocated as a large format sequential data set, larger than 64K tracks, and you are backing up to DASD, you must preallocate the journal backup data set as a large format sequential data set, which is large enough to contain a copy of the original large format sequential journal.

You do not have to be concerned about the size of the tape backup data sets, because DFSMShsm can span up to 18 tapes for control data set and journal data set backup copies.

Considerations for DFSMShsm control data sets and the journal

The following considerations apply to all DFSMShsm control data sets.

Backup considerations for the control data sets and the journal

The DFSMShsm control data sets and journal are backed up at the beginning of automatic backup processing. To prevent the control data sets and the journal from also being backed up as part of volume backup processing, ensure that the control data sets and journal are backed up only with the SETSYS CDSVERSIONBACKUP command.

If your control data sets are on SMS-managed volumes, ensure that they are backed up only with the CDSVERSIONBACKUP command (and not as part of user data set backup) by:

- Placing the control data sets on volumes whose storage group is defined with

  AUTO BACKUP ===> NO

  or

- Associating the control data sets with a management class whose attributes are

  AUTO BACKUP ===> N

Additionally, if you want to back up the control data sets using concurrent copy, ensure that the control data sets are associated with the BACKUP COPY TECHNIQUE management class attribute of REQUIRED(R), PREFERRED (P), VIRTUALREQUIRED (VR), VIRTUALPREFERRED (VP), CACHEREQUIRED (CR), or CACHEPREFERRED (CP).

If your control data sets are on non-SMS-managed volumes, ensure that they are backed up only with the CDSVERSIONBACKUP command (and not as part of user data set backup) by specifying the ALTERDS command.

Related reading:

- In addition to the considerations in this topic, you should also review "Improving performance of CDS and journal backup" on page 25.
- For more information about CDS and journal backup and using the SETSYS CDSVERSIONBACKUP command, see *z/OS DFSMShsm Storage Administration*.

Migration considerations for the control data sets and the journal

Ensure that the control data sets, control data set backup copies, journal data set, and journal data set backup copies are not allowed to migrate. Migration of any of the preceding data sets could make recovery impossible.

If your control data sets are on SMS-managed volumes, ensure that they do not migrate by:

- Placing them on volumes that are defined in a storage class with

  AUTO MIGRATION ===>NO

  or

- Associating the control data sets with a management class whose attributes are

  COMMAND OR AUTO MIGRATE ===> NONE

If your control data sets are on non-SMS-managed volumes, ensure that they do not migrate by specifying the NOMIGRATION parameter of the SETMIG command.

Refer to the following publications for more information about preventing data set migration in SMS and non-SMS environments:

- For SMS, see the topic about specifying migration attributes in *z/OS DFSMShsm Storage Administration*.
- For non-SMS, see the topic about controlling the migration of data sets and volumes in *z/OS DFSMShsm Storage Administration*.

Volume allocation considerations for the control data sets and the journal

This topic is very important because it defines the rules for creating the control data sets and journal.

When a control data set is defined on one volume, it is defined as a VSAM KSDS cluster. When the control data set is defined on two to four volumes, it is defined on each volume as a unique VSAM KSDS cluster. DFSMShsm can manage the control data set even if it consists of several VSAM clusters. Based on your calculations for the space required to define your control data set, you determine the number of volumes that are required and, therefore, the number of required clusters. The MCDS and BCDS may each be on more than one volume; the OCDS can be on only one volume.

Rules: Because extended addressability (EA) is a type of extended format VSAM data set that allows greater than 4 GB of data storage, use the following rules when you define a non-EA control data set:

1. A CDS cluster cannot be greater than one volume; therefore, a VSAM cluster must be defined for each CDS volume. The MCDS and BCDS may each be one–four clusters (volumes); the OCDS may be only one cluster (volume). When a CDS is defined with more than one cluster, it is referred to as a multicluster CDS.
2. The maximum size of one cluster is 4 GB.
3. No secondary allocation should be specified for a non-RLS cluster. To increase space, consider splitting the CDS up to four separate clusters. If you specify secondary allocation for a non-RLS CDS, be aware of the following:
   - Secondary allocation is allowed only for a single-cluster CDS.
   - A deadlock may occur.
   - The CDS monitor issues a message (ARC0909E or ARC0911E) each time the monitor threshold is reached. This means that if the CDS allocation can increase in size; then, for example, the 80% mark will also grow in size. So, as the CDS increases in size, the monitor message may be issued more than once.
   - ARC0130I will be issued when DFSMShsm is started. This message is intended to inform you that secondary allocation has been specified for this CDS.
4. The data component of one control data set must not be on the same volume as the index component of another control data set unless the index of the first control data set is also on the volume.
5. For performance reasons, index components of different control data sets should not be on the same volume.
6. Specify FREESPACE(0 0) to prevent misleading ARC0909E messages due to the way DFSMShsm updates and inserts records into control data sets and the method used to determine the percent full of a CDS.

   When calculating the percent full of a CDS, the numerator is the amount of space between the beginning of the data set and the high-used point (HURBA) of the data set. The denominator is the total space available in the data set, which is the amount of space between the beginning and the end of the data set (HARBA).

For a CDS, DFSMShsm does not subtract the free space below the high-used point because it can still exist when VSAM indicates the data set is full. For example, there can be free space in some control intervals and control areas below the high-used point in a key-sequenced data set (KSDS). However, an insert of a new logical record can still result in a return code indicating an out-of-space condition if there is no available space above the high-used point in the KSDS.

Space utilization in a VSAM KSDS is dependent on the location of a new record insert. For example, space must be free in the control intervals or a control interval must be free in the control area where VSAM performs the insert. Otherwise, VSAM obtains a new control area created after the high-used point to split the current control area.

To determine if there is embedded free space in a CDS, issue QUERY CONTROLDATASETS, which will display the percentage of free space.

The journal is restricted to one volume and must be allocated with contiguous space and without secondary allocation. Because the frequency of automatic backup of the control data sets and the journal varies from site to site, you should allocate 100 contiguous cylinders as the primary space allocation using SPACE=(CYL,(100),,CONTIG).

Ensure that the control data sets and journal data set are allocated on low-use mounted or private volumes. This allocation enhances the performance of DFSMShsm by reducing data set contention problems and by simplifying backup and recovery procedures for the control data sets.

If the control data sets and journal are SMS-managed, ensure that you assign them to a storage class with the GUARANTEED SPACE attribute.

Before you use the starter set for DFSMShsm, define volumes for the control data sets and journal data set. Use the following guidelines to determine which volumes to use for the control data sets and journal:

- You should not allocate the DFSMShsm control data sets on volumes containing JES3 data sets or system data sets. If they are allocated on such volumes, requests for JES3 data sets or system data sets could conflict with an DFSMShsm request for the control data sets, causing performance problems. A system lockout might occur in this situation.
- Each control data set must be allocated on a permanently resident volume. In a multiple DFSMShsm-host environment, these volumes must be shared when DFSMShsm is active on all processing units. The volume should be included in the SYS1.PARMLIB member, VATLST*xx*. VATLST*xx* is a volume attribute list defining the volumes mounted at system initialization with appropriate volume mount and use attributes. For more information about multiple DFSMShsm-host environment considerations, see "CDS considerations in a multiple DFSMShsm host environment" on page 269.
- The starter set uses the CDSVERSIONBACKUP parameter of the SETSYS command for more than one backup version of each control data set. The backup versions are directed to tape by the subparameter BACKUPDEVICECATEGORY(TAPE). If you choose to change this parameter to DASD, ensure that the backup data sets are preallocated on DASD volumes not

owned by DFSMShsm and ensure that those DASD volumes are in a storage group that is defined with AUTO BACKUP ===> NO and AUTO MIGRATION ===> NO.

When DFSMShsm is active (at least one DFSMShsm is operational), do not run any other application (other than DFSMShsm) that may access any control data set. The MCDS, BCDS, or OCDS must not be accessed by any other application while DFSMShsm is active. To allow another application to access the control data sets, DFSMShsm must be stopped.

RACF considerations for the control data sets and the journal

You should protect all control data sets with resource protection such as RACF, a component of the Security Server for z/OS. RACF protects the control data sets from being updated by unauthorized programs and unauthorized personnel.

For more information about protecting control data sets, see Chapter 9, “Authorizing and protecting DFSMShsm commands and resources,” on page 169.

Translating resource names in a GRSplex

For multiple HSMplexes in a single GRSplex, use the startup procedure keyword RNAMEDSN to specify if you want to upgrade your system to the new translation method. This new method allows DFSMShsm to translate minor global resource names to ones that are unique to one HSMplex. If you specify RNAMEDSN=YES, DFSMShsm translates minor resource names to unique values that avoid interference between HSMplexes. If you specify RNAMEDSN=NO, minor resource names remain compatible with all prior releases of DFSMShsm.

Determining the CDS serialization technique

Use the QUERY CONTROLDATASETS command to display the CDS serialization technique currently in use.

For more information about CDS record level sharing, see *z/OS DFSMShsm Storage Administration*.

Using VSAM record level sharing

DFSMShsm supports VSAM record level sharing (RLS) for accessing the control data sets. RLS enables DFSMShsm to take advantage of the features of the coupling facility for CDS access.

Accessing control data sets in RLS mode reduces contention when running primary space management and automatic backup on two or more processors. DFSMShsm benefits from the serialization and data cache features of VSAM RLS and does not have to perform CDS VERIFY or buffer invalidation.

Requirements for CDS RLS serialization

Control data sets accessed in RLS mode enqueue certain resources differently from control data sets accessed in non-RLS mode. The following are software and hardware requirements for CDS RLS serialization in a multiprocessor environment:

Software
: Global resource serialization or an equivalent function is required.

Hardware
: All operating systems running with DFSMShsm must be coupling facility capable (SP 5.2 and up), and the processors must have access to the coupling facility.

For more information about using VSAM record level sharing in a multiple DFSMShsm-host environment, see Chapter 11, "DFSMShsm in a multiple-image environment," on page 253.

Multiple host considerations

In installations where DFSMShsm hosts share the same control data sets, if one host on any z/OS image accesses the control data sets in RLS mode, all hosts on all z/OS images in the HSMplex must access the control data sets in RLS mode.

DFSMShsm will fail at startup if the selected serialization mode of a starting DFSMShsm host is incompatible with the CDS serialization mode of another DFSMShsm host that is already actively using the same control data sets.

Required changes to the CDS before accessing RLS

Before you can access the control data sets in RLS mode, they must be defined or altered to be RLS eligible. You must do this for all three control data sets (BCDS, MCDS, and OCDS). The control data sets must also be SMS-managed, using a storage class definition that indicates which coupling facility to use.

You must alter or define the control data sets using the LOG(NONE) attribute.

Example: ALTER *cdsname* LOG(NONE) . Whenever you alter or define the control data sets, do not use the ALL or UNDO parameters of the LOG keyword.

If it ever becomes necessary to change the control data sets back to non-RLS eligible, use the **ALTER** *cdsname* **NULLIFY(LOG)** command. Doing so requires that the SMSVSAM server is active. Otherwise, you must manually reset the catalog indicator for the control data sets.

To enable non-RLS read access to the control data sets when DFSMShsm is accessing them using RLS, you must define the control data sets with SHAREOPTIONS(2 3). Non-RLS read access includes functions such as EXAMINE and REPRO.

VSAM RLS coupling facility structures recommendations

It is suggested that a unique VSAM RLS cache structure be used for the DFSMShsm control data sets. See *z/OS DFSMSdfp Storage Administration* for information on sizing the cache and lock structures required for VSAM RLS support.

Required CDS version backup parameters

If CDS version backup is invoked while the control data sets are accessed in RLS mode, you must specify DFSMSdss as the datamover. Also, if the backup is directed to tape, the PARALLEL parameter must be used.

Invoking CDS RLS serialization

Use the startup procedure keyword CDSSHR with the RLS parameter to invoke RLS serialization. Whenever the control data sets are accessed in RLS mode, any values specified for CDSQ and CDSR are ignored. You can specify the CDSSHR keyword in the following manner:

CDSSHR = {YES | RLS | NO}

where

YES Performs multiprocessor serialization of the type requested by the CDSQ and CDSR keywords

RLS Performs multiple processor serialization using RLS

NO Does not perform multiple processor serialization

For a complete description of the CDSSHR keyword, see “CDSSHR” on page 304.

Using multicluster control data sets

Multicluster control data sets are control data sets that, because they are very large in size, require more than one volume to store their contents.

As your DFSMShsm workload increases so does the activity that each of the control data sets must record. As control data sets grow, they can require more space than the physical DASD devices allow (for example, the capacity of a 3390-3 is 2.8 GB). If this occurs, control data sets can be split across multiple volumes to accommodate their larger size. Only the MCDS and BCDS can be more than one cluster.

Considerations for using multicluster control data sets

Multicluster control data sets are a related group of VSAM clusters with the following requirements:

- The control data sets cannot be empty when you start DFSMShsm. The MCDS must be defined such that the high key for the first cluster must be greater than X'10' || C'MHCR'.
- All clusters in a multicluster CDS must be cataloged in the same user catalog.
- All clusters must be on DASD of the same category.
- Only primary allocation can be specified for the index and data; no secondary allocation can be specified for either the index or data.
- The index and data are limited to one volume; they cannot be on different volumes.
- For a multicluster BCDS, the RECORDSIZE and data CONTROLINTERVALSIZE for each cluster depend on the maximum number of backup versions you want to be able to keep for any data set.
- All key must be contiguous and must start with X'00' and end with X'FF'.
- The maximum record size must be the same for all clusters.
- The record key must start at 0 and be 44 characters in length.
- The CISIZE value must be the same for the data and index components for all clusters.
- The CONTROLINTERVALSIZE must be the same for all clusters of a multicluster CDS.
- The volume device type must be the same for all clusters. For example. if volumes are defined as shared, then all volumes must be defined as shared.
- Only primary allocation can be specified for non-RLS control data sets; no secondary allocation can be specified for non-RLS control data sets.

Message ARC0130I will be issued if the above rules are not followed.

Attention: Message ARC0264A is issued to confirm the new cluster count after the initialization of the first DFSMShsm host when the number of CDS clusters has been update. Verify the new cluster count indicated in the message is the intended number of clusters. If new cluster count is incorrect, enter N and verify the configuration changes to avoid CDS corruption.

New customers who have determined that their control data sets will span volumes when they fully implement DFSMShsm should do one of the following tasks:

1. Initially define each CDS on a single volume. As the MCDS and BCDS grow, you can reorganize them into multiple clusters in order to span volumes.
2. Define the control data sets as single cluster and multivolume. To do this, define the control data sets to use extended addressability (EA) and to use any of the following methods to access them: record level sharing (RLS), CDSQ serialization, or CDSR serialization.

For more information about control data set placement in a multiple DFSMShsm-host environment, see “CDS considerations in a multiple DFSMShsm host environment” on page 269.

Converting a multicluster control data set from VSAM key range to non-key-range

If you have multicluster control data sets that are defined as VSAM key ranges, then you need to convert those control data sets to not use VSAM key ranges. A good time to convert the control data sets to be non-key range is during a planned reorganization of the control data sets. To do the conversion, remove the key range keyword from the IDCAMS DEFINE statements that are used to define the multicluster control data sets. Then follow your normal reorganization process. When you restart DFSMShsm, because the VSAM key ranges are not present, DFSMShsm dynamically calculates the key boundaries of each cluster. You can use the QUERY CONTROLDATASETS command to view both the low and high keys that DFSMShsm calculated for each cluster.

Determining key ranges for a multicluster control data set

You need to determine where to split a CDS so that the data records in the CDS are evenly divided. DFSMShsm provides an application, SPLITCDS, that analyzes the current CDS data in all clusters and produces a report that shows the split ranges for two, three, and four clusters. You can analyze your existing control data sets using the HSM.SAMPLE.TOOL(SPLITCDS) job that is found in SYS1.SAMPLIB.

Figure 5 is an example of the typical key ranges for a two-cluster CDS.

```
BCDS:
     FROMKEY(X'00')         TOKEY(HSM.BACK.T2)
     FROMKEY(HSM.BACK.T3)   TOKEY(X'FF')

MCDS:
     FROMKEY( X'00')        TOKEY(HSM.HMIG.T4)
     FROMKEY(HSM.HMIG.T5)   TOKEY(X'FF')
```

Figure 5. Typical Key Ranges for a Two-cluster CDS

In Figure 5, the key range of the BCDS is split on the time stamp that indicates when the data set was backed up. A split using a single letter is not sufficient because the data components of the BCDS consist mostly of MCC records that share the same high-level qualifier. The high-level qualifier is identical for all MCC records (and therefore all backed up data sets) because it is taken from the SETSYS BACKUPPREFIX command. For more information about naming backup version (MCC) records, see *z/OS DFSMShsm Storage Administration*.

Figure 5 on page 35 also shows a split of the MCDS with keys X'00'–*prefix*.HMIG.T4 and keys *prefix*.HMIG.T5–X'FF' in cluster 2 where *prefix* is derived from the SETSYS MIGRATEPREFIX command and T*n* is derived from the time stamp in the MCA record.

Note:

1. HSM.SAMPLE.TOOL, which contains member SPLITCDS, is created by running ARCTOOLS, which resides in SYS1.SAMPLIB.
2. If the CDS is SMS-managed, ensure that the associated storage group contains the volumes that you have specified.

Multicluster control data set conversion

If the space required for your CDS is more than one volume, you need to split your CDS. The objective is to distribute the CDS data evenly across all of the new volumes. A VSAM cluster is defined for each volume. This topic guides you through converting to a multicluster CDS.

Steps for converting to a multicluster control data set:

Before you begin

You should determine the following before converting to a multicluster CDS:

- The amount of disk space required
- The number of volumes that are needed. Each volume will be a VSAM cluster
- The key range for each cluster. Refer to "Determining key ranges for a multicluster control data set" on page 35.

Procedure

1. Review "Considerations for using multicluster control data sets" on page 34.
2. Stop DFSMShsm on all z/OS images.
3. Back up the CDS you are converting using the access method services (AMS) EXPORT command.
4. Define a new multicluster CDS using the AMS DEFINE CLUSTER command.

   The following sample shows definitions for a multicluster (2 clusters) MCDS and BCDS.

```
//HSMCDS   JOB ,MSGLEVEL=(1,1)
//***************************************************************/
//* SAMPLE JCL THAT ALLOCATES MULTICLUSTER CONTROL DATA SETS.   */
//***************************************************************/
//*
//STEP1    EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSIN DD *
 DEFINE CLUSTER (NAME(DFHSM.MCDS1) -
     STORAGECLASS(SCLASS1) -
     CYLINDERS(2)  -
     RECORDSIZE(200 2040) FREESPACE(0 0) -
     INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
     UNIQUE LOG(NONE)) -
   DATA -
     (NAME(DFHSM.MCDS1.DATA) -
     CONTROLINTERVALSIZE(12288)) -
   INDEX -
     (NAME(DFHSM.MCDS1.INDEX) -
     CONTROLINTERVALSIZE(2048))
  DEFINE CLUSTER (NAME(DFHSM.MCDS2) -
     STORAGECLASS(SCLASS1) -
     CYLINDERS(2)  -
     RECORDSIZE(200 2040) FREESPACE(0 0) -
     INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
     UNIQUE LOG(NONE)) -
   DATA -
     (NAME(DFHSM.MCDS2.DATA) -
     CONTROLINTERVALSIZE(12288)) -
   INDEX -
     (NAME(DFHSM.MCDS2.INDEX) -
     CONTROLINTERVALSIZE(2048))
 DEFINE CLUSTER (NAME(DFHSM.BCDS1) -
     STORAGECLASS(SCLASS1) -
     CYLINDERS(2)  -
     RECORDSIZE(334 6544) FREESPACE(0 0) -
     INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
     UNIQUE LOG(NONE)) -
   DATA -
     (NAME(DFHSM.BCDS1.DATA) -
     CONTROLINTERVALSIZE(12288)) -
   INDEX -
     (NAME(DFHSM.BCDS1.INDEX) -
     CONTROLINTERVALSIZE(2048))
 DEFINE CLUSTER (NAME(DFHSM.BCDS2) -
     STORAGECLASS(SCLASS1) -
     CYLINDERS(2)  -
     RECORDSIZE(334 6544) FREESPACE(0 0) -
     INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
     UNIQUE LOG(NONE)) -
   DATA -
     (NAME(DFHSM.BCDS2.DATA) -
     CONTROLINTERVALSIZE(12288)) -
   INDEX -
     (NAME(DFHSM.BCDS2.INDEX) -
     CONTROLINTERVALSIZE(2048))
 DEFINE CLUSTER (NAME(DFHSM.OCDS1) -
     STORAGECLASS(SCLASS1) -
     CYLINDERS(2)  -
     RECORDSIZE(1800 2040) FREESPACE(0 0) -
     INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
     UNIQUE LOG(NONE)) -
   DATA -
     (NAME(DFHSM.OCDS1.DATA) -
     CONTROLINTERVALSIZE(12288)) -
   INDEX -
     (NAME(DFHSM.OCDS1.INDEX) -
     CONTROLINTERVALSIZE(2048))
```

5. Copy the old CDS to the new multicluster CDS with the access method services REPRO command.

The following sample JCL copies the old CDS into the new multicluster CDS.

```
/*
//******************************************************************/
//* COPY THE OLD CONTROL DATA SETS INTO THE NEWLY DEFINED          */
//*  MULTICLUSTER CONTROL DATA SETS.                               */
//* NOTE: THE FROMKEY/TOKEY VALUES ARE ONLY SAMPLES.  THE ACTUAL   */
//* PARAMETERS USED FOR THESE KEYWORDS SHOULD BE DERIVED FROM      */
//* ACTUAL CDSS BEING USED.                                        */
//******************************************************************/
//*
//STEP2    EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSIN DD *
 REPRO INDATASET(DFHSM.MCDS) OUTDATASET(DFHSM.MCDS1) -
    FROMKEY(X'00') TOKEY(MIDDLE.KEY1)
 REPRO INDATASET(DFHSM.MCDS) OUTDATASET(DFHSM.MCDS2) -
    FROMKEY(MIDDLE.KEY2) TOKEY(X'FF')
 REPRO INDATASET(DFHSM.BCDS) OUTDATASET(DFHSM.BCDS1) -
    FROMKEY(X'00') TOKEY(MIDDLE.KEY1)
 REPRO INDATASET(DFHSM.BCDS) OUTDATASET(DFHSM.BCDS2) -
    FROMKEY(MIDDLE.KEY2) TOKEY(X'FF')
 REPRO INDATASET(DFHSM.OCDS) OUTDATASET(DFHSM.OCDS1)
/*
```

6. Modify the DFSMShsm startup procedure in SYS1.PROCLIB and any other JCL (such as DCOLLECT and ARCIMPRT) that references the multicluster CDS. There must be a separate DD card for each cluster of a multicluster CDS. For more information, see "Updating the startup procedure for multicluster control data sets" on page 39 and "Updating the DCOLLECT JCL for multicluster control data sets" on page 39.
7. Preallocate new CDS backup data sets if you back up your control data sets to DASD. You need backup versions for each cluster in the CDS.

   Note: Do not delete the current CDS. Instead, maintain it for a period of time until you determine that the new CDS is valid.
8. Monitor the growth of the multicluster CDS.

Changing the number of clusters of a multicluster control data set

Whenever you need to increase or decrease the number of clusters for a multicluster CDS, you can perform nearly the same steps as you did to initially convert to multicluster usage. Follow the steps in "Steps for converting to a multicluster control data set" on page 36 for converting to a multicluster control data set. As you next restart DFSMShsm, the new key boundaries that are used in your new clusters are dynamically determined and message ARC0087I is issued to prompt you to perform a CDS version backup. You should promptly back up the CDS because a CDS recovery requires that the last existing key boundaries as recorded in the journal match those in the CDS backup copy.

Attention: Message ARC0264A is issued to confirm the new cluster count after the initialization of the first DFSMShsm host when the number of CDS clusters has been update. Verify the new cluster count indicated in the message is the intended number of clusters. If new cluster count is incorrect, enter N and verify the configuration changes to avoid CDS corruption.

Changing only the key boundaries of a multicluster control data set

If, however, it is necessary to redistribute the key boundaries but not to change the number of clusters, then perform the following additional step. Patch the number of clusters in the MHCR record for the appropriate CDS to X'FF' immediately before shutting DFSMShsm down for the reorganization. A value of X'FF' in the number of clusters field signals to DFSMShsm that the user is intentionally modifying the key boundaries; users can immediately create the appropriate CDS backup to maintain the recoverability of the control data sets. If you have not invalidated the number of clusters but restart DFSMShsm after the reorganization, DFSMShsm issues message ARC0130 RC=13, and then it shuts down. This is done to prevent the CDS from becoming irrecoverable should an error occur.

To change the key boundaries in a multicluster CDS, perform the following steps:

1. Issue a FIXCDS command to set the one byte field in the MHCR that contains the number of clusters for the appropriate CDS to X'FF'.
2. After invalidating the number of CDS cluster in the MHCR record, shut down DFSMShsm and follow the steps in "Steps for converting to a multicluster control data set" on page 36 for converting to a multicluster control data set. As you restart DFSMShsm, it dynamically determines the key boundaries that are in your new clusters and it issues message ARC0087I to prompt you to perform the CDS version backup. Perform the CDS version backup immediately after you restart DFSMShsm. Figure 6 shows sample FIXCDS commands that you can use to clear the number of clusters in a multicluster CDS.

```
FIXCDS S MHCR PATCH(X'159' X'FF') /* MCDS */
FIXCDS S MHCR PATCH(X'15A' X'FF') /* BCDS */
```

Figure 6. Sample FIXCDS Commands to Invalidate Number of Clusters in Multicluster CDS

Updating the startup procedure for multicluster control data sets

The DD cards that define the control data sets must be updated to indicate to DFSMShsm that the control data sets are multicluster. Table 2 shows the DD names for single-cluster control data sets and the associated DD names for multicluster control data sets.

Table 2. Associated DD Names for MIGCAT and BAKCAT Single-Cluster and Multicluster Control Data Sets

Rename single-cluster CDS	To
MIGCAT	MIGCAT, MIGCAT2, MIGCAT3, MIGCAT4
BAKCAT	BAKCAT, BAKCAT2, BAKCAT3, BAKCAT4

If CDS clusters are defined incorrectly (for example, overlaps or gaps in the key ranges), DFSMShsm issues message ARC0130I describing the violation.

Updating the DCOLLECT JCL for multicluster control data sets

The DD cards that define the control data sets must be updated to indicate to DCOLLECT that the control data sets are multicluster. Table 3 shows the DD names for single-cluster control data sets and the associated DD names for multicluster control data sets.

Table 3. Associated DD Names for MCDS and BCDS Single-Cluster and Multicluster Control Data Sets

Rename single-cluster CDS	To
MCDS	MCDS, MCDS2, MCDS3, MCDS4
BCDS	BCDS, BCDS2, BCDS3, BCDS4

Using VSAM extended addressability capabilities

DFSMShsm supports VSAM KSDS extended addressability (EA) capabilities that use any of the following serialization techniques for accessing its control data sets: record level sharing (RLS) access mode, CDSQ serialization, or CDSR serialization. VSAM EA capabilities allow each cluster to exceed the 4 GB size. The MCDS and BCDS can span up to four unique KSDS clusters. The OCDS is limited to a single cluster. The same serialization technique must be used to access all control data sets.

Rules: Use the following rules when you use CDS extended addressability:

CDS EA in RLS mode
: Specify **CDSSHR=RLS** for all DFSMShsm systems sharing the same control data sets.

CDS EA with CDSQ serialization
: Specify the **CDSQ=YES** and **CDSR=NO** for all DFSMShsm systems sharing the same control data sets.

CDS EA with CDSR serialization
: Specify the **CDSQ=NO** and **CDSR=YES** for all DFSMShsm systems sharing the same control data sets.

CDS EA with CDSQ and CDSR serialization
: Specify the **CDSQ=YES** and **CDSR=YES** for all DFSMShsm systems sharing the same control data sets.

Multicluster control data sets
: Multicluster control data sets can still use CDSQ, CDSR, or CDSQ and CDSR. They cannot be defined with key ranges, but must use dynamic allocation support.

Requirements: The following requirements may affect your use of extended addressability for your control data sets:

- Mixing extended function (EF) clusters and non-EF clusters is permissible because each cluster is treated as a separate entity. However, if any cluster is accessed in RLS mode, then all clusters must be accessed in RLS mode.
- Examine your disaster recovery plans to ensure that your disaster recovery site can support the use of EA control data sets with the proper serialization. You can use CDSQ serialization, CDSR serialization, or RLS.

Note:

1. You can define each CDS as EA or non-EA; however, you do not have to define all control data sets as EA or non-EA.
2. You can define each MCDS or BCDS cluster as EA or non-EA.
3. Extended addressability control data sets can be single cluster and multivolume or multicluster and multivolume.
4. The MCDS and the BCDS can be represented by up to 4 KSDS EA data sets. Separate clusters reduce backup processing time by allowing parallel operations. In addition, if only one cluster is damaged or lost, the recovery time to forward recover this one cluster would be reduced.

Converting control data sets to extended addressability with either CDSQ or CDSR serialization

You can convert your control data sets to extended addressability for use with either CDSQ or CDSR serialization. You must ensure that the EA control data sets are SMS-managed and are assigned to a DATA CLASS that specifies extended format and extended addressability.

The allocation of the EA CDS and the copying of the contents of the existing CDS is a one time process. Use the IDCAMS DEFINE CLUSTER command to allocate the EA CDS. Then use the IDCAMS REPRO command to copy the contents of the existing CDS to the newly allocated CDS.

DFSMShsm problem determination aid facility

The problem determination aid (PDA) facility gathers sufficient DFSMShsm processing information to pinpoint module flow and resource usage that is related to any DFSMShsm problem. The PDA facility is required for IBM service because it traces module and resource flow. DFSMShsm stores its trace information in the PDA log data sets.

DFSMShsm accumulates problem determination information at specific module points in the form of trace data, and it records this data in main storage. At predetermined intervals, the trace data is scheduled for output to DASD. The DFSMShsm trace recording function receives the trace data that is scheduled for output and writes this data to a file on DASD. The PDA facility consists of two separate log data sets. DFSMShsm recognizes these log data sets by their DD names, ARCPDOX and ARCPDOY. Recording takes place in the data set defined by ARCPDOX. When that data set is filled, the two data set names are swapped, and recording continues on the newly defined data set.

When this data set is filled, the names are again swapped, and the output switches to the other data set, thus overlaying the previously recorded data. The larger the data sets, the longer the period of time that is represented by the accumulated data.

The preferred implementation of the PDA facility is to establish a protocol that automatically copies the ARCPDOY data set to tape as a generation-data-group data set each time message ARC0037I is issued. This practice provides a sequential history of trace data over time so that the data is available when needed for resolving problems.

When PDA trace is activated, the DFSMShsm default is to trace all events. Conditional tracing can be specified with a PATCH command to deactivate certain traces. This reduces the amount of PDA tracing that is done. See “Running conditional tracing” on page 364 for additional information.

Problem determination aid log data sets

PDA log data sets provide you with trace information about DFSMShsm processing.

Problem Determination Aid Log Data Sets

Required:
No, but strongly recommended.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical sequential.

Storage guidance:
Both the ARCPDOX and ARCPDOY data sets must be on the same volume.

The amount and type of storage you use for your PDA log data sets depends on how much trace history you want to keep. To determine the amount and type of storage, you can use either the short-term work sheet found in "Problem determination aid log data set size work sheet—Short-term trace history" on page 390 or the long-term work sheet found in "Problem determination aid log data set size work sheet—Long-term trace history" on page 392.

Planning to use the problem determination aid (PDA) facility

Before you can use the PDA facility, you need to:

1. Determine how long you want to keep trace information.
2. Allocate storage on DASD for the PDA log data sets, ARCPDOX and ARCPDOY.
3. Implement the PDA facility based on how long you want to keep trace information.

Determining how long to keep trace information

How many hours, days, or weeks of trace history does your site want to keep? The minimum recommended trace history is four hours; however, a longer trace history gives a greater span both forward and backward in time. Your choice of a trace history interval falls into one of the following categories:

Short-term trace history
One to two days is typically considered a short-term trace history interval. Short-term trace histories can be obtained without using generation data groups (GDGs).

Long-term trace history
Two or more days is typically considered a long-term trace history interval. Long-term trace histories are best implemented with the use of generation data sets (GDSs) appended sequentially to form a generation data group (GDG).

A long-term trace history is preferred because some DFSMShsm processing (weekly dumps for instance) occurs only on a periodic basis. The longer the trace history, the more obvious is the context within which you can perform problem analysis.

Problem determination aid (PDA) log data set size requirements

The amount of storage that you allocate for your PDA log data sets depends on the amount of trace data activity at your site and depends on how long you want to keep trace information. As part of your considerations for "Planning to use the problem determination aid (PDA) facility," you may have considered how long you want to keep trace information.

Short-term PDA trace history
: If you choose to keep trace information for two days or less, see "Problem determination aid log data set size work sheet—Short-term trace history" on page 390.

Long-term PDA trace history
: If you choose to keep trace information for longer than two days, see "Problem determination aid log data set size work sheet—Long-term trace history" on page 392.

Controlling the problem determination aid (PDA) facility

The problem determination aid (PDA) facility is automatically enabled during DFSMShsm startup. You can enable or disable PDA processing by the way you specify the SETSYS PDA(NONE|ON|OFF) command.

If you specify . . .	Then . . .
SETSYS PDA(NONE) during DFSMShsm startup	No dynamic storage is obtained, the DASD trace data sets are not opened, and no data is gathered.
SETSYS PDA(ON)	DFSMShsm requests storage for data accumulation and opens the DASD trace data sets if they have been allocated. If no DASD trace data sets have been allocated, the data is accumulated only in internal storage.
SETSYS PDA(OFF)	All data accumulation is halted, but the DASD trace data set remains open.

You should continuously enable PDA tracing when DFSMShsm is active; any resulting performance degradation is minimal.

The PDA log data sets are automatically swapped at DFSMShsm startup. There is no way to control swapping at startup, but there can be later times when you may want to switch the data sets before ARCPDOX is filled. To switch the data sets, use the SWAPLOG PDA command.

The *z/OS DFSMShsm Diagnosis* contains additional information about the PDA facility, or see "Problem determination aid log data set size work sheet—Short-term trace history" on page 390.

Note: If the DFSMShsm PDOX data set is on a non-SMS-managed volume that will be backed up, use the ALTERDS *datasetname* VERSIONS(0) command to prevent DFSMShsm from backing it up.

If the PDOX data set is on an SMS-managed volume that will be backed up, assign it to an SMS management class that does not allow backup copies.

Allocating the problem determination aid (PDA) log data sets

Figure 7 on page 44 shows how to allocate and catalog the problem determination log data sets. These data sets have been allocated for a single DFSMShsm-host as part of the starter set.

```
/*********************************************************************/
/* SAMPLE JOB THAT ALLOCATES AND CATALOGS THE PDA LOG DATA SETS.      */
/*********************************************************************/
/*
//ALLOPDO JOB MSGLEVEL=1,TYPRUN=HOLD
//STEP1   EXEC PGM=IEFBR14
//DD1     DD DSN=&UID..?HOSTID..HSMPDOX,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//           VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
//DD2     DD DSN=&UID..?HOSTID..HSMPDOY,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//           VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
/*
```

Figure 7. Sample JCL Job that Allocates and Catalogs the PDA Log Data Sets

Change the User ID (?UID.), the processing unit ID (?HOSTID.),the trace unit (?TRACEUNIT.), and the volume serial number (?TRACEVOL.) parameters to names that are valid for your environment. The LRECL and RECFM fields will be set by DFSMShsm when the data set is opened and are not required in the JCL. These data sets must be variable blocked physical sequential and must not be striped. Both data sets must be allocated to the same volume. If you allocate them as SMS-managed data sets they must be associated with a storage class having the GUARANTEED SPACE attribute. They should not be associated with a storage class that will conflict with the required data set attributes. If you are using the starter set, the two DD statements (DD1 and DD2) that you need are already allocated.

Attention: If you allocate the problem determination log data sets as SMS-managed data sets, they must be associated with a management class that prohibits backup and migration. If you allocate them as non-SMS-managed data sets, they must be on a volume that DFSMShsm does not process for backup or migration.

Printing the problem determination aid (PDA) log data sets

For information about printing the problem determination aid logs, refer to *z/OS DFSMShsm Diagnosis*.

DFSMShsm logs

DFSMShsm provides feedback to the storage administrator and system programmer with the following DFSMShsm logs:

Log	Description
DFSMShsm log	DFSMShsm logs are data sets named LOGX and LOGY in which DFSMShsm writes processing statistics and system event statistics. The logs are useful for monitoring, tuning, and verifying your storage management policies. The DFSMShsm log provides information on processing-unit events and processing statistics.
Edit log	The edit log depends on the DFSMShsm log for its input and edits the DFSMShsm log to provide specialized reports.
Activity logs • Backup • Dump • Migration • Aggregate backup and recovery • Command	The activity logs report on the backup, dump, migration, ABARS, and command processing of DFSMShsm in your system.

DFSMShsm log data set

The DFSMShsm log data sets provide DFSMShsm with information about events on a particular processing unit and about commands that are entered with the LOG command. The DFSMShsm log records this information in chronological order. However, the SMF and the problem determination aid data sets already record much of the information in the log. If you already are keeping SMF and PDA data and you do not have an ISV product that needs to directly scan the LOGX/LOGY files, or if you do not intend to use the HSMLOG procedure, it is recommended that you do not maintain LOGX/LOGY files. Specify DD DUMMY and use the HOLD LOG command.

DFSMShsm Log Data Set

Required:
: No.

Allocated by starter set:
: Yes, see "Starter set example" on page 109.

Data set type:
: Two physical sequential data sets.

Storage guidance:
: The two physical sequential DFSMShsm log data sets must be on the same volume.

Two physical sequential data sets, known to the starter set as HSMLOGX and HSMLOGY, together make up the DFSMShsm log. DFSMShsm records information in the LOGX data set until the LOGX data set is full. Then DFSMShsm swaps the LOGX data set with the LOGY data set, exchanges the names, and informs the operator that DFSMShsm has swapped the log data sets.

To automatically swap the DFSMShsm log data sets at startup, specify LOGSW=YES on the PROC statement of your DFSMShsm startup procedure. You can also swap the DFSMShsm log data sets with the SWAPLOG command. If you do not want the log data sets to swap at startup, specify LOGSW=NO. To prevent the current log from being overwritten during startup, specify DISP=MOD in the DD statements for the log data sets.

The data set names for LOGX and LOGY are not required to be any particular names, but can be originated by your site. Each log entry is stored as a logical record in the physical sequential log data sets. DFSMShsm begins each log entry record with a log record header that contains the record length, record type, creation time, and creation date. The log entry records include:

- Function statistics records (FSR), consisting of statistical data about the functions performed by DFSMShsm. The FSR includes start and stop times for DFSMShsm functions and is updated after each data set that DFSMShsm processes.
- Daily statistics records (DSR), containing DFSMShsm processing statistics. The processing statistics are updated every hour (if the data has changed) that DFSMShsm is active during the current day.
- Volume statistics records (VSR), consisting of one record for each volume DFSMShsm processes during the current day. The record is updated each hour (if the data has changed) that DFSMShsm is active.

- Time, when DFSMShsm receives a data set or volume request and when each request is completed. Each request contains a management work element (MWE).
- DFSMShsm error processing description (ERP) and subtask abnormal end data (ESTAI).
- Data entered with the LOG command.

For the formats of the FSR, DSR, and VSR records, see *z/OS DFSMShsm Diagnosis*.

Should a system outage occur that does not close the log, DFSMShsm protects against the loss of the log data by periodically saving the address of the written record.

DFSMShsm log size

The amount of space that you allocate for the two DFSMShsm log data sets depends on the amount of DFSMShsm activity at your site and how often you want to swap and print the logs. Initially, on a 3390 device, you should allocate two cylinders as the primary space allocation and one cylinder as the secondary space allocation: SPACE=(CYL,(2,1)). If your log data sets are automatically swapped too frequently, increase the primary and secondary space allocation.

Optionally disabling logging

Because the information stored in the DFSMShsm log is duplicated elsewhere, such as SMF, PDA, and activity logs, it is recommended that you disable the log to improve performance, unless you need the DFSMShsm log for external reporting, such as from third-party products. If you choose not to use logging, use one of the following methods:

- Specify a DD DUMMY statement in the startup procedure where HSMLOGX and HSMLOGY are allocated.
- Do not specify any DD statements for log data sets.
- Specify the HOLD LOG command in the DFSMShsm startup procedure.

Printing the DFSMShsm log

z/OS DFSMShsm Storage Administration contains information about printing the DFSMShsm log. It also contains sample lists from the ARCPRLOG and ARCPEDIT programs.

Edit log data sets

The edit log provides you with selected information that can be edited from the LOGY data set of the DFSMShsm log. These selected records are the result of running the HSMLOG procedure that is described in detail in "HSMLOG procedure" on page 311.

Edit Log Data Set

Required:
No.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical sequential.

Printing the edit log

z/OS DFSMShsm Storage Administration contains information about printing the edit log. It also contains sample lists from the ARCPRLOG and ARCPEDIT programs.

Activity log data sets

Activity log data sets provide you with messages that relate to activity in one of five areas: space management, backup and recovery, dumps, ABARS, and command functions of DFSMShsm.

Activity Log Data Sets

Required:
No.

Allocated by starter set:
No, DFSMShsm dynamically allocates the activity log data sets.

Data set type:
Physical sequential or SYSOUT.

General guidance:
SETSYS ACTLOGMSGLVL and ABARSACTLOGMSGLVL control the amount and kind of generated messages.

SETSYS ACTLOGTYPE and ABARSACTLOGTYPE control where messages are written. **Example:** You can direct messages to a printer or to a DASD file where they can then be browsed.

See "Starter set example" on page 109 for an example of how these commands are specified in the starter set.

DFSMShsm has five activity logs, each providing information that relates to activity in one of four DFSMShsm areas.

Log	Description
Migration Activity Log	Reporting on space management activity, this log is closed either when the volume space management control module finishes processing or when the storage administrator issues the RELEASE HARDCOPY command. The RELEASE HARDCOPY command has no effect for logs that are empty. This processing includes MIGRATE commands for volumes and levels, interval migration, automatic primary space management, and automatic secondary space management.
Backup Activity Log	Reporting on automatic backup or volume command backup, FRBACKUP and FRRECOV activities, and volume command recovery activities, this log is closed either after automatic backup has ended, after a set of BACKVOL, RECOVER, or EXPIREBV commands has been processed, or when the storage administrator issues the RELEASE HARDCOPY command. The RELEASE HARDCOPY command has no effect for logs that are empty.
Dump Activity Log	Reporting on automatic dump or command volume dump and command volume restore activities, this log is closed either after automatic dump has ended, after a set of BACKVOL or RECOVER commands has been processed, or when the storage administrator issues the RELEASE HARDCOPY command. The RELEASE HARDCOPY command has no effect for logs that are empty.

Log	Description
ABARS Activity Log	Reporting on aggregate backup and aggregate recovery activities, this log is closed after each aggregate backup, and each aggregate recovery request has been completed. One ABARS activity log is allocated for each aggregate backup or aggregate recovery command issued. This log is allocated, opened, written to, and closed by the ABARS secondary address space The SETSYS ABARSDELETEACTIVITY command allows you to specify whether you want DFSMShsm to automatically delete the ABARS activity log associated with an aggregate backup or recovery version during roll off or EXPIREBV ABARSVERSIONS processing.
Command Activity Log	This log reports on TAPECOPY and TAPEREPL activity, and also records error informational messages that occur during low-level type internal service processing. This log is closed either when DFSMShsm shuts down or when the storage administrator issues the RELEASE HARDCOPY command. The RELEASE HARDCOPY command has no effect for logs that are empty.

Activity log information for the Storage Administrator

The migration, backup, dump, and ABARS activity logs contain high-level messages about DFSMShsm activity with which storage administrators can monitor DFSMShsm activity.

Activity log information for the System Programmer

The command activity log contains detailed messages with which a system programmer determines system error conditions. Each message is date-stamped and time-stamped indicating when the message was issued. A header and trailer message indicates the span of time that is covered by the log.

The activity logs are not the same as the DFSMShsm log that is used for maintenance. The activity logs verify that DFSMShsm is performing as you expect.

The migration, backup, and dump activity logs receive records for automatic volume functions and for command volume functions. Each receives:

- A record for the start of each automatic function
- A record for the end of each automatic function
- A record for the start of each volume
- A record for the end of each volume

In addition, the backup and migration logs can receive an ARC0734I message for each data set that is processed.

Controlling the amount of information written to the activity logs

The activity logs can provide a large amount of information, making it time-consuming to examine each log. You can reduce the time that is required to examine the logs in two ways: by controlling the amount of information written to the logs and by providing for online analysis of the logs.

If you specify . . .	Then . . .
SETSYS ACTLOGMSGLVL	DFSMShsm controls the amount of information that is written to all activity logs, except the ABARS activity logs. All ABARS messages are always written to the ABARS activity log.

If you specify . . .	Then . . .
SETSYS ABARSACTLOGMSGLVL	DFSMShsm controls the amount of information written to the ABARS activity logs. Both FULL and REDUCED are allowed on this command. However, EXCEPTIONONLY is not supported.
SETSYS ACTLOGMSGLVL (FULL)	All eligible records are written to the backup, dump, and migration logs.
SETSYS ACTLOGMSGLVL (REDUCED)	DFSMShsm logs message ARC0734I for each new DFSMShsm-owned data set that is created when data sets are successfully migrated from level-0 volumes by space management or when data sets are successfully copied (or scheduled to be copied) from level-0 volumes by backup. The ARC0734I message is also written when data sets are successfully copied from migration volumes during backup of migrated data sets.

The REDUCED option removes complexity from the analysis of the activity logs by suppressing messages about successful moves of DFSMShsm-owned data sets between DFSMShsm's owned devices and for certain cleanup kinds of data set deletions. Data set movements and deletions for which successful ARC0734I messages are not issued are:

- Movement of a migrated data set to another migration volume
- Extent reduction
- Scratching of utility, list, and temporary data sets
- Movement of existing backup versions (recycle, move backup copies, and spill processing)
- Scratching of not valid migration copies
- Scratching of not valid backup versions
- Deletion of aged control data set records for statistics and for formerly migrated data sets

The ARC0734I message is always recorded for any *unsuccessful* attempt to move, copy, or delete a data set.

Note: You can cause message ARC0734I to be issued for a data set that is determined by DFSMShsm to be unqualified for selection during SMS-managed volume migration or SMS-managed volume backup. The reason code that is issued with the message explains why the data set did not qualify for selection. For more information about using the PATCH command to initiate this error message, refer to *z/OS DFSMShsm Diagnosis*.

If you specify . . .	Then . . .
SETSYS ACTLOGMSGLVL (EXCEPTIONONLY)	DFSMShsm writes the ARC0734I data set message only when a message returns a nonzero return code. A nonzero return code indicates that an error occurred in processing a data set. **Tip:** When volumes are being dumped, either automatically or by command, set the ACTLOGMSGLVL parameter to FULL or REDUCED to obtain a list of all data sets that are dumped. After dumping has completed, you can set the parameter back to EXCEPTIONONLY.

Controlling the device type for the activity logs

The second way you can reduce the burden of analyzing the activity logs is to automate the process. The ABARSACTLOGTYPE and the ACTLOGTYPE parameters allow you to choose whether to make the activity logs SYSOUT data sets or DASD data sets. If you send the activity logs to a DASD data set, you can browse the data online for information that is of interest to you.

DFSMShsm dynamically allocates DASD data sets with a unit name of SYSALLDA and a size of 20 tracks for primary allocation or 50 tracks for secondary allocation. Activity logs have names in the following forms:

Type of Activity Log	Activity Log Name
ABARS	*mcvtactn*.H*mcvthost*.*function*.*agname*.D*yyddd*.T*hhmmss*
All Other Types	*mcvtactn*.H*mcvthost*.*function*.D*yyddd*.T*hhmmss*

mcvtactn
: Activity log high-level qualifier

H, D, and T
: Constants

mcvthost
: Identifier for the DFSMShsm host that creates these activity logs

function
: ABACKUP, ARECOVER, CMDLOG, BAKLOG, DMPLOG, or MIGLOG

agname
: Aggregate group name

yyddd
: Year and day of allocation

hhmmss
: Hour, minute, and second of allocation

Note:

1. If you do not want the activity log data sets to appear in the master catalog, define an alias for HSMACT.
2. DFSMShsm provides a high-level qualifier of HSMACT for *mcvtactn*. If your data set naming convention is not compatible with this qualifier, you can use a PATCH command to modify this high-level qualifier. See “Replacing HSMACT as the high-level qualifier for activity logs” on page 334.

Considerations for creating log data sets

The information in this topic applies to the various DFSMShsm log data sets.

Ensure that the active DFSMShsm log and the active problem determination log are not allowed to migrate. To prevent migration of data sets, see *z/OS DFSMShsm Storage Administration*.under the heading “Controlling Migration of Data Sets and Volumes” for non-SMS-managed data sets, or under the heading “Specifying Migration Attributes” for SMS-managed data sets.

Before you can start DFSMShsm with the starter set, you need to define a volume for the various DFSMShsm log data sets. You also need to perform the following tasks:

- Ensure that the log data sets are allocated on low-use mounted or private volumes to enhance the performance of DFSMShsm by reducing data set contention.
- Ensure that the various DFSMShsm log data sets can be SMS managed so you can assign them to a storage class with the GUARANTEED SPACE attribute.

DFSMShsm small-data-set-packing data set facility

The small-data-set-packing (SDSP) data set facility of DFSMShsm allows DFSMShsm to migrate small user data sets from level-0 volumes and to store them as records on level-1 migration volumes where the records of several data sets can share the same track. Small user data sets are stored in SDSP data sets on level-1 migration volumes.

If a data set is eligible to be migrated to an SDSP data set, DFSMShsm selects a migration level-1 volume containing a non-full SDSP data set. If no migration level-1 volumes contain a usable SDSP data set, the data set migrates as a separate data set. If the data set is ineligible for migration to an SDSP data set, it migrates as a data set. SDSP data sets are allocated only on migration level-1 volumes. Each migration level-1 volume can contain only one SDSP data set.

Figure 8 on page 53 illustrates how small-data-set-packing data sets save space on level-1 migration volumes. SDSP data sets offer the following advantages:

- Reduced fragmentation of a level-1 volume.
- Reduced use of space in the volume table of contents (VTOC). Because the small-user data sets are stored as records, migration level-1 VTOCs are *not* filled with an entry for each small user data set that resides in an SDSP.
- Better use of space on the migration level-1 volumes. Small data sets become records in the SDSP data sets and, as such, share the same tracks.

Small-Data-Set-Packing Data Sets

Required:
No.

Allocated by starter set:
Yes, see “ALLOSDSP” on page 145.

Maximum record size:
2093 bytes.

Data set type:
VSAM key-sequenced data set (KSDS) on level-1 migration volume.

Storage guidance:
SDSP data sets must reside on level-1 DASD migration volumes, and each must have a specific name. Allocate only one SDSP data set per level-1 migration volume.

Preparing to implement small-data-set-packing data sets

To begin using the SDSP facility, you first need to perform the following tasks:

- "Defining the size of a small user data set"
- "Allocating SDSP data sets"
- "Specifying the SDSP parameter on the ADDVOL statement"

Defining the size of a small user data set

You define the size of a "small" user data set to DFSMShsm when you specify the SMALLDATASETPACKING parameter of the SETSYS command. You can define small data sets in either KB (KB = 1024 bytes) or tracks of a 3380 volume. DFSMShsm calculates the size requirements in KB for each potentially eligible user data set and compares the size to the number you have specified.

You should use a setting of 150KB. Data sets smaller than 150KB are considered small enough to be eligible for SDSP processing.

Partitioned data sets (PDS) cannot migrate to SDSP data sets.

Allocating SDSP data sets

The amount of storage that you allocate for your SDSP data sets depends on the space that can be attributed to small data sets on your DFSMShsm-managed volumes. **Rule:** Only one SDSP data set is allowed on each volume.

After you have chosen a size for your small data sets, measure the amount of storage that is presently used for data sets of that size.

Specifying the SDSP parameter on the ADDVOL statement

Ensure that you specify the SDSP parameter on the ADDVOL command for any level-1 migration volumes on which you allocate an SDSP data set.

Data mover considerations for SDSP data sets

Any data set that is supported by the DFSMSdss data mover is considered for migration to an SDSP. For a listing of data sets types that are supported by data mover DFSMSdss and data mover DFSMShsm, review the tables that are located in "Supported data set types" on page 61.

VSAM considerations for SDSP data sets

The SDSP data set is a VSAM data set and must be primed and initialized before you can use it. Remember that SDSP data sets require periodic reorganization, as do any other VSAM key-sequenced data sets. You can use the access method services EXPORT and IMPORT commands to reorganize the SDSP data sets of your computing system at regular intervals.

You can significantly reduce the need to reorganize SDSP data sets by enabling the CA reclaim function for them. For more information, see the topic about reclaiming CA space in *z/OS DFSMS Using Data Sets*.

The optimum data control interval size (CISIZE) for an SDSP data set residing on a 3390 DASD is 26 624, which allows you to write 360 2093-byte records per data control area (cylinder).

Figure 8 shows an overview of small-data-set-packing data sets.

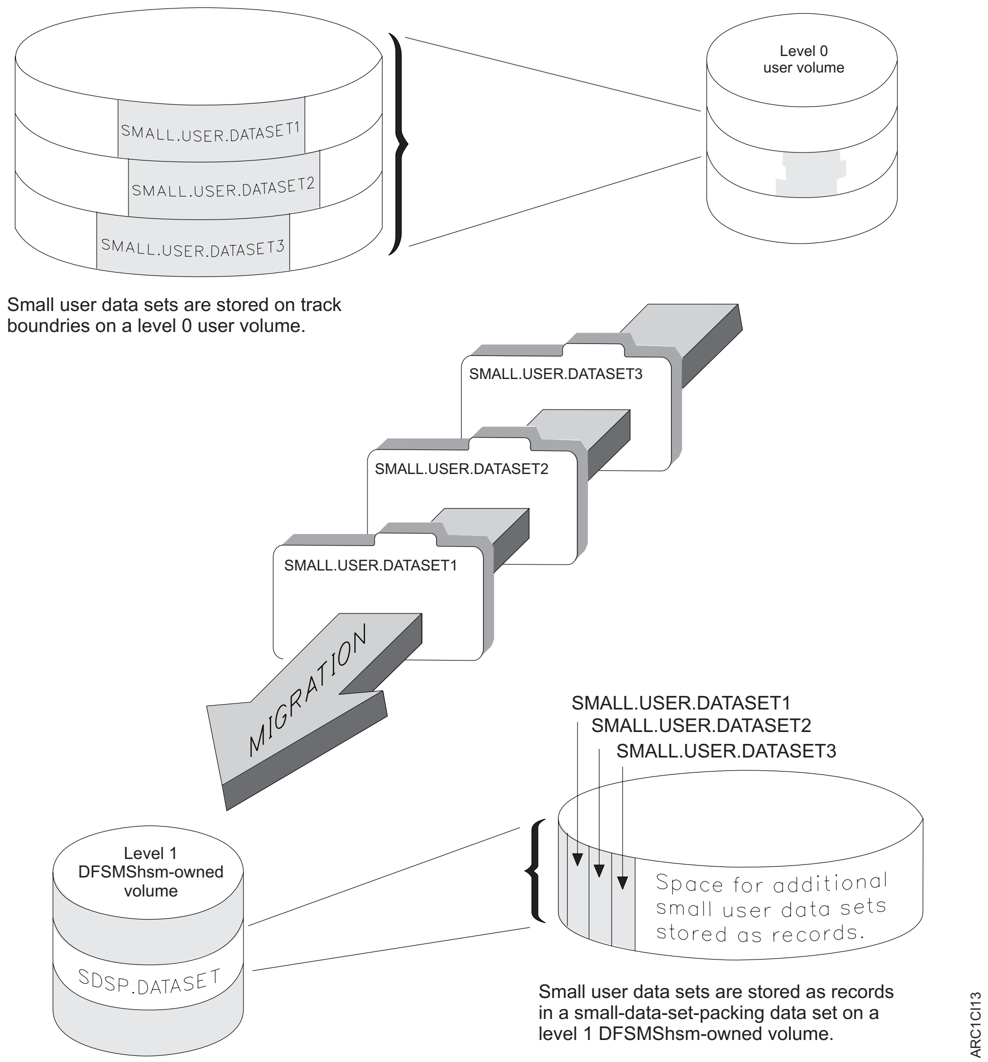


Figure 8. Small-Data-Set-Packing Data Set Overview

Small user data sets reside on separate tracks of level-0 user volumes. DFSMShsm can migrate the small user data sets (as records) and store them as records in a VSAM key-sequenced SDSP data set on a level-1 migration volume. DASD space is reduced because multiple data sets then share the same tracks on level-1 migration volumes.

Multitasking considerations for SDSP data sets

Though one SDSP data set can be used for each concurrent migration task, there are some DFSMShsm activities that have a higher usage priority for SDSP data sets. These activities are:

- Recall processing
- Aggregate backup processing
- FREEVOL processing
- AUDIT MEDIACONTROLS processing
- Automatic secondary space management processing

Figure 9 shows the potential resource contention that exists in the SDSP environment:

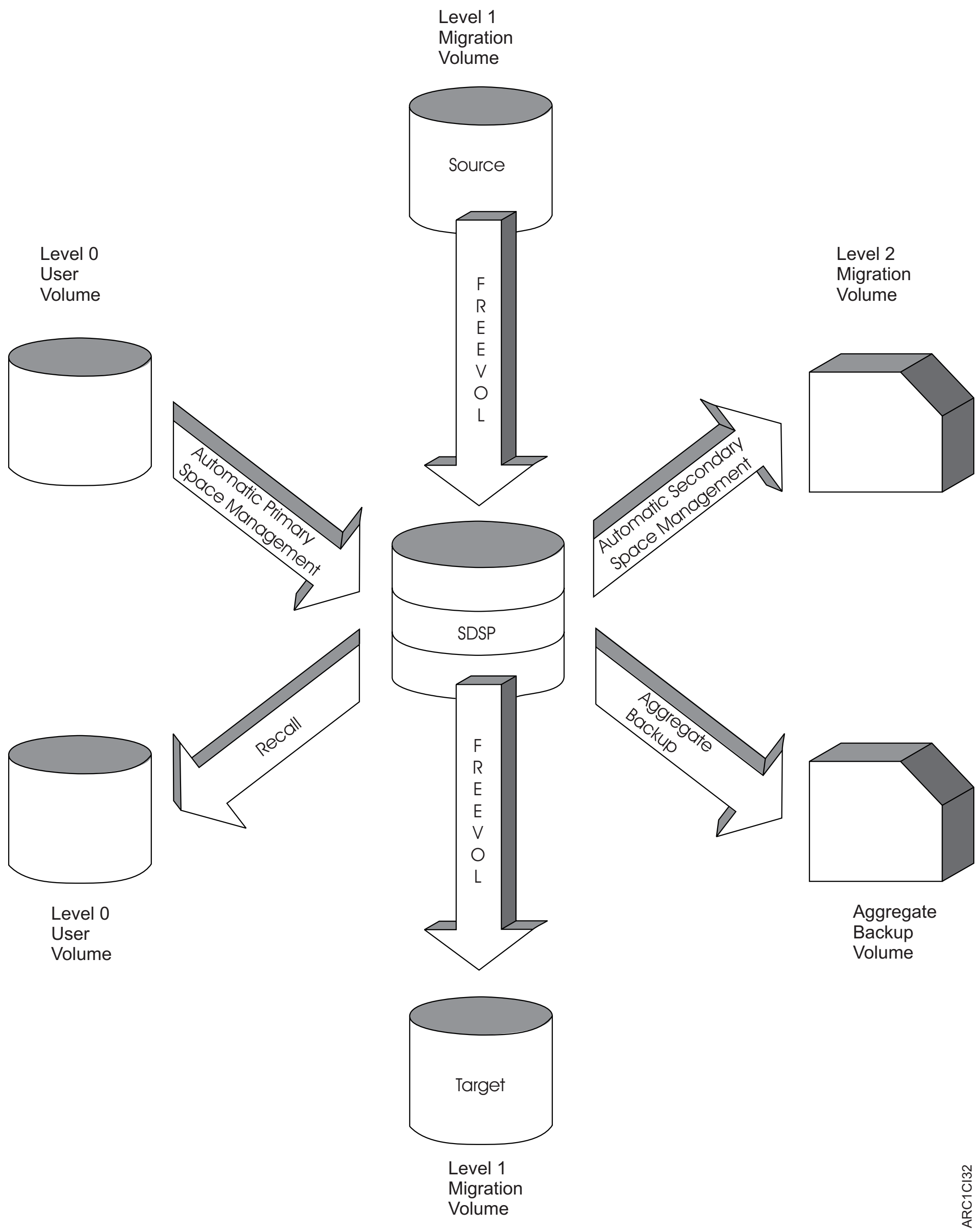


Figure 9. The SDSP Data Set Contention Environment

It is important to plan the number of SDSP data sets in relation to the number of concurrent migration tasks and the amount of processing done by functions with a higher usage priority for the SDSP data sets.

Because of their higher usage priority, any of these activities can gain control of your SDSP data sets and leave you with fewer than the expected number of SDSP data sets for migration.

When an activity with a higher usage priority for SDSP data sets has or requests an SDSP data set, that SDSP data set is no longer a candidate for migration. The small data set that is in need of migration must find a different, available SDSP data set or it is skipped and left unprocessed until your next migration window.

Additionally, if all SDSP data sets should become full (as a result of migrations to them), the filled SDSP data sets are not candidates for further migration. Full SDSP data sets are not seen by migration processing, and, as a result, any small user data sets are migrated as data sets to level-1 migration volumes.

The following three-part example illustrates how SDSP data sets become unavailable for use as level-0 to level-1 migration targets. Figure 10 shows three concurrent migration tasks that move three small user data sets from level-0 user volumes to level-1 migration volumes (with SDSP data sets that are defined on the level-1 migration volumes).

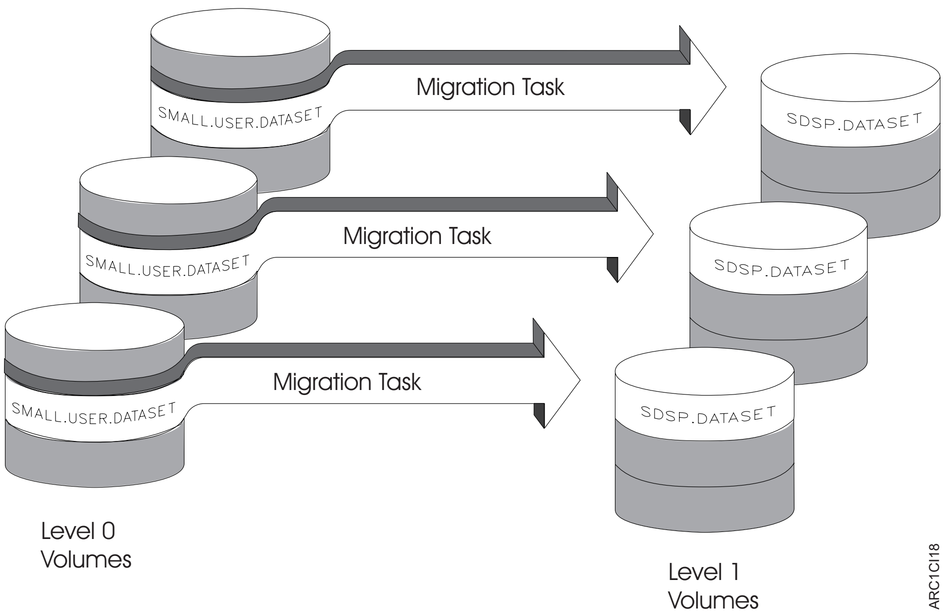


Figure 10. Part 1—Ideal Multitasking Migration. Three migration tasks migrate three small user data sets to three level-1 migration volumes on which SDSP data sets are defined.

Figure 11 on page 56 shows how a recall of a small user data set from an SDSP data set during level-0 to level-1 migration has effectively eliminated one concurrent migration task. The small user data set, whose migration was preempted by a recall, sees that all SDSP volumes are not full and defers this migration for your next migration window.

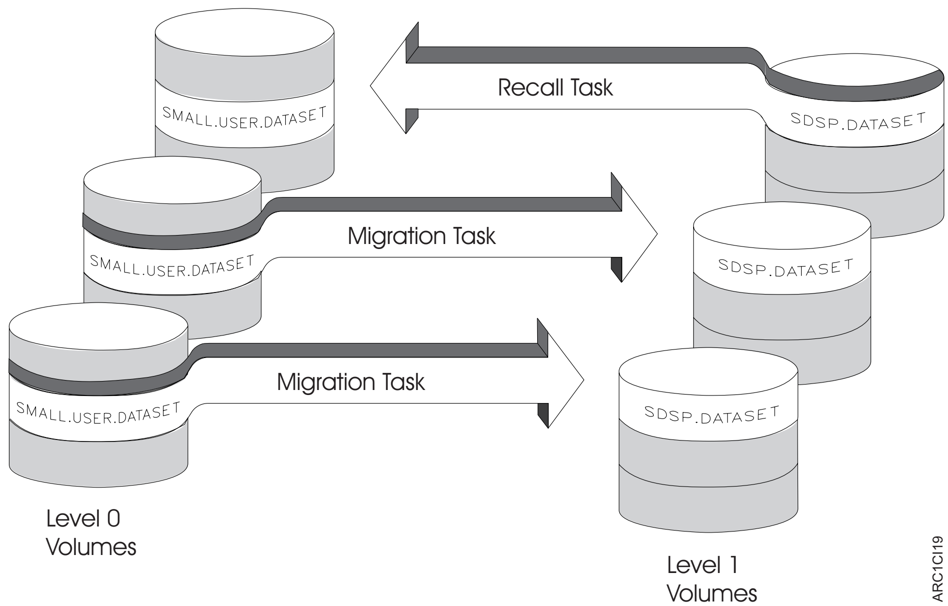


Figure 11. Part 2—Recall Processing Has a Higher Priority than Migration. One migration task does not process the small data sets because recall processing has a higher usage priority for the SDSP than migration processing.

Figure 12 shows that all SDSP data sets have become full. They are no longer seen as candidates for level-0 to level-1 migration destinations and the small-user data sets migrate as data sets.

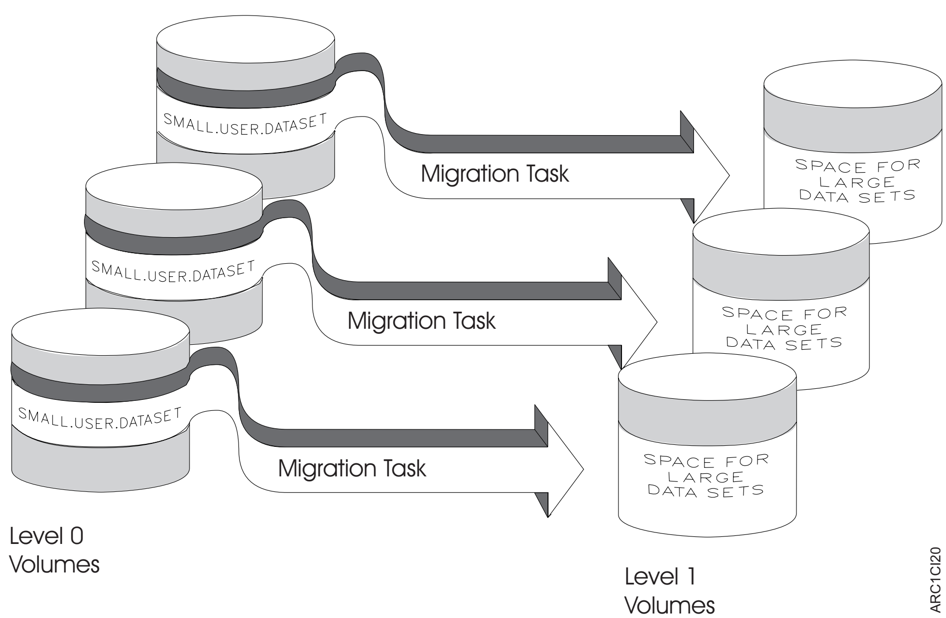


Figure 12. Part 3—All SDSP Data Sets Are Full. The small user data sets migrate as large data sets to the level-1 migration volumes.

Because other activity can effect your migrations, you must plan and monitor those activities that can cause your small user data set migrations to be skipped. You must define ample SDSP data sets to manage your worst-case scenario.

Related reading: For more information about the SDSP parameter of the SETSYS command and for a table of SDSP migration contention priorities, see *z/OS DFSMShsm Storage Administration*.

System data sets

The following data sets are the data sets that DFSMShsm uses to interact with various MVS facilities. The MSYSIN and MSYSOUT data sets are dummy data sets that DFSMShsm uses to support TSO messages and batch processing.

Data set with DDNAME of MSYSIN

This data set provides DFSMShsm with a DUMMY SYSIN statement for DFSMShsm support of TSO batch processing. DFSMShsm needs this data set for the system services that TSO invokes on behalf of DFSMShsm.

MSYSIN Data Set

Required:
Yes.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical Sequential.

Data set with DDNAME of MSYSOUT

This data set provides DFSMShsm with the messages that are issued by the terminal monitor program and with the messages that are issued when dynamic memory allocation takes place. DFSMShsm needs this data set for the system services that TSO invokes on behalf of DFSMShsm.

MSYSOUT Data Set

Required:
No.

Allocated by starter set:
Yes, see "Starter set example" on page 109.

Data set type:
Physical Sequential.

Chapter 4. User data sets

The data sets that are discussed here are data sets that the DFSMShsm product manages—user data sets. Data set types that are supported by data mover DFSMShsm and data mover DFSMSdss are also discussed.

User data sets supported by DFSMShsm

DFSMShsm space management and backup functions support the following data set organizations when accessed by the appropriate standard MVS access methods:

- Physical sequential (PS) — including large format data sets
- Partitioned (PO)
- Partitioned data set extended (PDSE)
- Direct access (DA)
- Virtual storage (VS)
- Hierarchical file system data sets (HFS)
- zFS data sets.

The dump functions of DFSMShsm support all data sets supported by the physical dump processes of DFSMSdss. For more information, refer to *z/OS DFSMSdss Storage Administration*.

If the data set organization field (DSORG) of the data set VTOC entry indicates PS, PO, DA, or VS access methods, DFSMShsm initiates management of the data set. Be aware that some application program processes are dependent on how the PS and DA data sets are blocked or accessed. Applications that do not use standard access methods for data sets may fail after the data has been processed by DFSMShsm. An example of this would be data sets that are created for IMS™ databases and accessed with the IMS overflow sequential access method (OSAM).

Physical sequential data sets

DFSMShsm moves (migrate/recall or backup/recovery) physical sequential (PS) data sets between unlike DASD device types automatically or by command and fully utilizes all tracks except the last. Requesting DFSMShsm to reblock PS Data sets is common, particularly when moving them between unlike DASD device types.

DFSMShsm determines whether checkpointed MVS and IMS GSAM SMS physical sequential data sets are eligible for migration. For checkpointed data sets, DFSMShsm migration, including extent reduction, is delayed for a fixed number of days. The default delay is five days. A checkpointed data set is eligible for migration when the date-last-referenced, plus the number of days the data set is treated as unmovable, are less than or equal to the current date. If the data set is eligible, DFSMShsm builds the command sequence for a DFSMSdss logical data set dump, including the keyword and parameter FORCECP(0). For command migration, error messages are issued for ineligible data sets.

DFSMShsm does not allow users to include the FORCECP keyword as part of the DFSMShsm MIGRATION command. Instead, see “Modifying the number of elapsed days for a checkpointed data set” on page 363 for information about a patch byte to use to force migration.

Physical sequential data sets and OSAM

Overflow sequential access method (OSAM) requires all data blocks, including the last one, to be the same length. DFSMShsm can manage data sets processed with OSAM when the DSORG field specifies PS (hereafter called PS/OSAM) if the PS/OSAM data sets are not reblocked.

Note: If this restriction is inadvertently bypassed, the data is not lost but can be addressed by OSAM after the data is returned to its original format.

If reblocking occurs, the last data block may be short. Therefore, the user must ensure that data can be addressed in PS/OSAM data sets by making certain that DFSMShsm does not reblock them.

If reblocking of a PS/OSAM data set occurs, a separate job must be run to return the data set to its original block size. The DFSMShsm data set reblock exit can be used to prevent reblocking of PS/OSAM data sets if their names can be identified.

Direct access data sets and OSAM

Overflow sequential access method (OSAM) requires full-track utilization. DFSMShsm can manage data sets processed with OSAM when the DSORG field specifies DA (hereafter called DA/OSAM) if the DA/OSAM data sets are not directed to a different device type either by command or by JCL.

Note: If this restriction is inadvertently bypassed, the data is not lost but can be addressed by OSAM after the data has been returned to its original format.

Therefore, the user must ensure that data in DA/OSAM data sets can be addressed by making certain that the data is not directed to a different device type. If a DA/OSAM data set is moved to a different device type, DFSMShsm commands should be used to migrate the data set and direct its recall back to the original device type. DFSMShsm does not specifically direct data sets to SMS-managed volumes.

Direct access data sets

When DFSMShsm selects a device type for recall or recovery of non-SMS direct access (DA) data sets, it selects the same device type as the original level 0 device type. DFSMShsm ensures the addressability of the recalled or recovered data by moving a track image copy of the data.

However, a user can issue a DFSMShsm command to direct a data set to a device type that has a smaller, larger, or the same track size. When the user directs the data, however, the addressability of that data depends on how the user tells DFSMShsm to move the data (relative track or relative block addressing) and the access method used by the application program. For more detailed information about recall or recovery of (DA) data sets, refer to *z/OS DFSMShsm Storage Administration*.

Hierarchical file system data sets

DFSMShsm processes hierarchical file system (HFS) data sets. HFS data sets contain a complete file system; however, DFSMShsm does not process individual files within a file system.

zFS data sets

DFSMShsm processes zFS data sets. zFS data sets are VSAM linear data sets (LDS) that provide a function similar to HFS data sets. DFSMShsm does not process individual file systems within a zFS data set.

Exceptions to the standard MVS access methods support

DFSMShsm attempts to process data sets with invalid formats. If an error is detected, the operation fails, and an error message is issued. If DFSMShsm does not detect an error, the results of a DFSMShsm operation are unpredictable and could result in deterioration of the condition of the data set. The following data sets can cause unpredictable results:

- Partitioned data sets with invalid directory block entries
- Variable-length-format data sets with embedded, nonstandard records
- Data sets with embedded physical blocks with a size not consistent with the block size indicated in the data set VTOC entry.

  Attention: With DFSMSdss as the data mover, uncataloged user data sets can be lost if the user tries to direct recovery of a cataloged data set with the same name to the same volume on which the uncataloged data set resides.

Size limit on DFSMShsm DASD copies

Whatever data mover is used, DFSMShsm uses the DFSMSdfp DADSM function to allocate storage on migration or backup DASD volumes. If a data set is larger than the available free space on one DASD backup or migration volume, the backup or migration will fail.

Supported data set types

Data set recovery with FRRECOV supports all data sets that DFSMSdss physical copy supports, with the following exceptions: User catalogs, VVDS, VTOC Index, VSAM Key-range, Migrated data sets, and GDG bases. VSAM components cannot be recovered individually. VSAM data sets must be recovered as a cluster.

This section contains tables that describe the supported data set types.

Data set type support for space management functions

Table 4 provides information about supported data set types for space management functions.

Table 4. Space Management—Data Set Type Support

Data Set Type	Volume Space Management DBA or DBU	Volume Space Management Expire or Delete	Volume Space Management	Command Data Set Migration
Nonintegrated catalog facility catalogs	NO	NO	NO	NO
Data sets whose names are SYSCTLG	NO	NO	NO	NO
Integrated catalog facility catalog	NO	NO	NO	NO
Uncataloged data sets	YES	YES	NO	NO

Table 4. Space Management—Data Set Type Support (continued)

Data Set Type	Volume Space Management DBA or DBU	Volume Space Management Expire or Delete	Volume Space Management	Command Data Set Migration
Cataloged data sets that are not accessible through the standard catalog search (these appear to DFSMShsm as uncataloged data sets)	NO	NO	NO	NO
VSAM data sets cataloged in existing catalogs (those not in the integrated catalog facility catalog)	NO	NO	NO	NO
Partitioned data sets (PO) with zero block size	NO	NO	NO	NO
Partitioned data sets (PO) with non-zero block size	YES	YES	YES	YES
Partitioned data sets (SMS-managed) having an AX cell in the VTOC/VVDS	NO	NO	YES	YES
Non-VSAM (non-SMS-managed) data sets on multiple volumes	NO	NO	NO	NO
Non-VSAM (SMS-managed) data sets on multiple volumes	NO	NO	YES	YES
Non-VSAM (SMS-managed) data sets on multiple volumes that are RACF indicated	NO	YES	NO	NO
Direct access (BDAM) data sets on multiple volumes	NO	NO	NO	NO
Split-cylinder data sets	NO	NO	NO	NO
User-labeled data sets that are empty	YES	YES	NO	NO
User-labeled data sets that are not sequential	NO	NO	NO	NO
User-labeled data sets that are sequential and not empty	YES	YES	YES	YES
Unmovable data sets with one extent	NO	NO	NO	NO
Unmovable data sets with more than one extent	NO	NO	NO	NO
Absolute track data sets (ABSTR)	NO	NO	NO	NO
Any data set allocated to another user with a disposition of OLD	NO	NO	NO	NO
List and utility data sets (but not SMS-managed)	YES	YES	NO	YES
List and utility data sets (SMS-managed)	YES	YES	YES	YES
Data sets whose names begin with HSM or SYS1 (except SYS1.VVDS)	NO 1	NO 1	NO 1	NO 1
1 This restriction can be removed using the SETMIG LEVEL() command.				
Data sets with no extents	NO	NO	NO	NO
Data sets cataloged with an esoteric unit name (for example, D3390 on DASD)	NO	NO	NO	NO
Authorized program facility (APF) authorized library	NO	NO	NO	NO
Password-protected generation data sets (use the information under the heading "Generation data groups" on page 379 regarding password-protected generation data sets, and for a method of bypassing this restriction)	NO	NO	NO	NO
Fully qualified generation data group names	YES	YES	YES	YES
ALIAS names	NO	NO	NO	YES 2
2 An ALIAS can be used with the HMIGRATE, HRECALL, and HDELETE commands.				
Partitioned data sets with more than one NOTE list or with more than three user TTRs per member	NO	NO	NO	NO

Table 4. Space Management—Data Set Type Support (continued)

Data Set Type	Volume Space Management DBA or DBU	Volume Space Management Expire or Delete	Volume Space Management	Command Data Set Migration
Extended partition data sets	NO	NO	YES	YES
Physical sequential data sets cataloged in ICF catalogs without special or unusual characteristics (multivolume, user labels, and so forth)	YES	YES	YES	YES
Physical sequential variable blocked data sets with a logical record length (LRECL) larger than block size (BLKSIZE)	YES	YES	YES	YES
Physical sequential large format data set	YES	YES	YES	YES
VSAM (non-SMS-managed) data sets whose components reside on more than one volume	NO	NO	NO	NO
VSAM (SMS-managed) data sets whose components reside on more than one volume	NO	NO	YES	YES
VSAM (non-SMS-managed) data sets defined with key ranges	NO	NO	NO	NO
VSAM (SMS-managed) data sets defined with key ranges	NO	NO	YES	YES
VSAM data sets with the page-space attributes	NO	NO	NO	NO
VSAM open data sets	NO	NO	NO	NO
VSAM data sets with the erase-on-scratch bit set in the catalog	NO	NO	NO	NO
VSAM data sets with the erase-on-scratch bit set in the RACF profile	YES	YES	YES	YES
VSAM data sets with 2 to 17 AIXs **3**	NO	NO	NO	YES
VSAM data sets with more than one path name associated with the AIX **3**	NO	NO	NO	YES
VSAM data sets whose base cluster has more than one path name **3**	NO	NO	NO	YES
3 If VSAM data sets have more than one AIX, more than one path, or more than one path to the AIX are migrated, all components except the base cluster name are uncataloged. These data sets must be recalled by using the base cluster name.				
VSAM data sets with more than 17 AIXs	NO	NO	NO	YES
VSAM (SMS-managed) data sets with an empty data or index component	NO	YES	YES	YES
VSAM (non-SMS-managed) data sets with an empty data or index component	YES	NO	YES	YES
VSAM data sets whose LRECLs are too long to be processed by EXPORT For relative record data sets (RRDS), the maximum length is 32 752 bytes. For entry-sequenced data sets (ESDS) and key-sequenced data sets (KSDS), the maximum length is 32 758 bytes.	YES	YES	YES	YES
VSAM spheres marked as forward recovery required	YES	YES	NO	NO
VSAM spheres with retained locks	NO	NO	NO	NO
Mounted zFS data sets	NO	NO	NO	NO
Unmounted zFS data sets	YES	YES	YES	YES

Data set type support for availability management functions

Table 5 provides information about data set type support for availability management functions.

Table 5. Availability Management—Data Set Type Support

Data Set Type	Aggregate Backup INCLUDE Processing	Aggregate Backup ALLOCATE Processing	Volume Backup	Command Data Set Backup
Nonintegrated catalog facility catalogs	NO	NO	YES	YES
Data sets whose names are SYSCTLG	NO	NO	YES	YES
Integrated catalog facility catalog	NO	YES	YES	YES
Uncataloged data sets	NO	NO	YES	YES
Cataloged data sets that are not accessible through the standard catalog search (these appear to DFSMShsm as uncataloged data sets)	NO	NO	NO	NO
VSAM data sets cataloged in existing catalogs (those not in the integrated catalog facility catalog)	NO	NO	NO	NO
Partitioned data sets (PO) with zero block size	YES	YES	NO	NO
Partitioned data sets (PO) with non-zero block size	YES	YES	YES	YES
Non-VSAM (non-SMS-managed) data sets on multiple volumes	YES	YES	NO	NO
Non-VSAM (SMS-managed) data sets on multiple volumes	YES	YES	YES	YES
Non-VSAM data sets on multiple volumes that are RACF indicated	YES	YES	NO	NO
Direct access (BDAM) data sets on multiple volumes	1	1	NO	NO
1 These data sets are not fully supported. They can be restored to multiple volumes only as non-SMS-managed. They can be restored to a single SMS volume by using the DFSMSdss patch described in *z/OS DFSMSdfp Storage Administration*.				
Split-cylinder data sets	YES	YES	NO	NO
User-labeled data sets that are empty	YES	YES	NO	NO
User-labeled data sets that are not sequential	YES	YES	NO	NO
User-labeled data sets that are sequential and not empty	YES	YES	YES	YES
Unmovable data sets with one extent	YES	YES	YES	YES
Unmovable data sets with more than one extent	YES	YES	NO	NO
Absolute track data sets (ABSTR)	YES	YES	YES	YES
Any data set allocated to another user with a disposition of OLD	YES	YES	YES 2	NO
2 These data sets can be backed up if allowed by the SETSYS BACKUP(INUSE(...)) subparameters or the installation exit ARCBDEXT, or both.				
List and utility data sets (but not SMS-managed)	YES	YES	NO	NO
List and utility data sets (SMS-managed)	YES	YES	YES	YES
Data sets whose names begin with HSM or SYS1 (except SYS1.VVDS)	YES	YES	YES	YES
Data sets with no extents	YES	YES	YES	YES
Data sets cataloged with an esoteric unit name (for example, D3390 on DASD)	YES	YES	YES	YES
Authorized program facility (APF) authorized library	YES	YES	YES	YES

Table 5. Availability Management—Data Set Type Support (continued)

Data Set Type	Aggregate Backup INCLUDE Processing	Aggregate Backup ALLOCATE Processing	Volume Backup	Command Data Set Backup
Password-protected generation data sets (use the information under the heading "Generation data groups" on page 379 regarding password-protected generation data sets and for a method of bypassing this restriction)	YES	YES	YES	YES
Fully qualified generation data sets (for example, X.Y.G0001V00)	YES	YES	YES	YES
Relative generation data sets with zero or negative generation numbers (for example, X.Y(-1))	YES	YES	NO	NO
Generation data group names	NO	NO	NO	NO
ALIAS names	NO	NO	NO	YES **3**
3 You can use ALIASes with the HBACKDS and HRECOVER commands. But when a NONVSAM data set with ALIASes is backed up, DFSMShsm does not save the ALIASes, so by using the HBACKDS and HRECOVER commands, the ALIASes are not rebuilt and are lost.				
Partitioned data sets with more than one NOTE list or with more than three user TTRs per member	YES	YES	NO	NO
Extended partition data sets and APF libraries	YES	YES	YES	YES
Physical sequential variable blocked data sets with a logical record length (LRECL) larger than block size (BLKSIZE)	YES	YES	YES	YES
Physical sequential data sets cataloged in ICF catalogs without special or unusual characteristics (multivolume, user labels, and so on)	YES	YES	YES	YES
Physical sequential large format data set	YES	YES	YES	YES
VSAM (non-SMS-managed) data sets whose components reside on more than one volume	YES	YES	NO	NO
VSAM (SMS-managed) data sets whose components reside on more than one volume	YES	YES	YES	YES
VSAM (non-SMS-managed) data sets defined with key ranges	YES	YES	NO	NO
VSAM (SMS-managed) data sets defined with key ranges	YES	YES	YES	YES
VSAM data sets with the page-space attributes	NO	YES	NO	NO
VSAM open data sets	YES	YES	YES **2**	YES **4**
4 These data sets can be backed up using inline backup (ARCINBAK).				
VSAM backup-while-open candidates	YES	YES	YES	YES
VSAM RLS-accessed data sets	YES	YES	YES	YES
VSAM data sets with the erase-on-scratch bit set in the catalog	YES	YES	YES	YES
VSAM data sets with the erase-on-scratch bit set in the RACF profile	YES	YES	YES	YES
VSAM data sets with 2 to 17 AIXs	YES	YES	YES	YES
VSAM data sets with more than one path name associated with the AIX	YES	YES	YES	YES
VSAM data sets whose base cluster has more than one path name	YES	YES	YES	YES
VSAM (SMS-managed) data sets with an empty data or index component	NO	YES	YES	YES

Table 5. Availability Management—Data Set Type Support (continued)

Data Set Type	Aggregate Backup INCLUDE Processing	Aggregate Backup ALLOCATE Processing	Volume Backup	Command Data Set Backup
Note: When a SMS VSAM cluster is backed up, its entire sphere is backed up; that is, all components and all associations including AIXs.				
VSAM (non-SMS-managed) data sets with an empty data or index component	YES	NO	YES	YES
VSAM data sets whose LRECLs are too long to be processed by EXPORT For relative record data sets (RRDS), the maximum length is 32 752 bytes. For entry-sequenced data sets (ESDS) and key-sequenced data sets (KSDS), the maximum length is 32 758 bytes.	YES	YES	YES	YES
Tape data sets in the INCLUDE list and data sets in the ALLOCATE list that have a BLKSIZE greater than 32 760 bytes	YES	NO	Not Applicable	Not Applicable
Tape data sets in the INCLUDE list and data sets in the ALLOCATE list that have an LRECL greater than 32 760 bytes	NO	NO	Not Applicable	Not Applicable
Tape data sets created by the DFSMSdss COPYDUMP function using the DFSMSdss defaults for the DCB information (LRECL = 0)	NO	NO	Not Applicable	Not Applicable
Data sets in the Allocate list that have a BLKSIZE greater than 32 760 bytes	NO	NO	Not Applicable	Not Applicable
Tape data sets in the INCLUDE list that have a BLKSIZE less than 524 288 bytes	YES	YES	Not Applicable	Not Applicable
Mounted zFS data sets	YES	NO	YES	YES
Unmounted zFS data sets	YES	NO	YES	YES

Chapter 5. Specifying commands that define your DFSMShsm environment

You can specify SETSYS commands in the ARCCMD*xx* member to define your site's DFSMShsm environment. The command options are described along with the reasons for choosing a command.

The starter set creates a basic (and somewhat generic) DFSMShsm environment. If you choose not to begin with the starter set or you want to expand or customize the starter set functions, the information you need is in this section.

Regardless of the DFSMShsm functions you choose to implement, you must establish the DFSMShsm environment for those functions. Your site's DFSMShsm environment is established when you perform the following tasks:

- "Defining the DFSMShsm startup environment"
- "Defining storage administrators to DFSMShsm" on page 74
- "Defining the DFSMShsm MVS environment" on page 75
- "Defining the DFSMShsm security environment for DFSMShsm-owned data sets" on page 83
- "Defining data formats for DFSMShsm operations" on page 86
- "Defining DFSMShsm reporting and monitoring" on page 90
- "Defining the tape environment" on page 92
- "Defining the installation exits that DFSMShsm invokes" on page 92
- "Controlling DFSMShsm control data set recoverability" on page 92
- "Defining migration level 1 volumes to DFSMShsm" on page 93
- "Defining the common recall queue environment" on page 95
- "Defining common SETSYS commands" on page 98

Defining the DFSMShsm startup environment

Before starting DFSMShsm, you must prepare the system by performing the following tasks:

- "Allocating DFSMShsm data sets"
- "Establishing the DFSMShsm startup procedures" on page 68
- "Establishing the START command in the COMMNDnn member" on page 71
- "Establishing SMS-related conditions in storage groups and management classes" on page 71
- "Writing an ACS routine that directs DFSMShsm-owned data sets to non-SMS-managed storage" on page 71
- "Directing DFSMShsm temporary tape data sets to tape" on page 72
- "Establishing the ARCCMDxx member of a PARMLIB" on page 73

Allocating DFSMShsm data sets

The DFSMShsm data sets are the data sets DFSMShsm requires for full-function processing. The DFSMShsm data sets are not user data sets and they are not DFSMShsm-managed data sets. Rather, they are the following DFSMShsm record keeping, reporting, and problem determination data sets:

- DFSMShsm control data sets
- DFSMShsm control data set copies
- Journal data set
- Log data sets
- Problem determination aid (PDA) log data sets
- SDSP data sets (if used)

You, or the person who installed DFSMShsm on your system, probably have allocated these data sets during installation or testing of DFSMShsm. The data sets are required for the DFSMShsm starter set. For SMS environments, you must associate the DFSMShsm data sets with a storage class having the GUARANTEED SPACE=YES attribute so that you can control their placement. Data sets having the guaranteed space attribute are allocated differently than non-guaranteed space data sets, especially if candidate volumes are specified. Refer to *z/OS DFSMShsm Storage Administration* for a discussion of the guaranteed space attribute and for information about establishing storage classes.

You must prevent the following DFSMShsm data sets from migrating:
- Control data sets
- DFSMShsm log data sets
- Journal
- Problem determination aid logs

For more information about preventing DFSMShsm data sets from migrating, see "Storage guidance for control data set and journal data set backup copies" on page 28 and "Migration considerations for the control data sets and the journal" on page 29.

Establishing the DFSMShsm startup procedures

If you specify an HSMPARM DD, it will take precedence over MVS concatenated PARMLIB support. However, if you are using MVS concatenated PARMLIB support, DFSMShsm uses the PARMLIB data set containing the ARCCMD*xx* member and the (possibly different) PARMLIB data set containing the ARCSTR*xx* member (if any) that is indicated in the startup procedure.

When ABARS is used, its address space (one or more) is termed 'secondary' to a 'primary address space'. That primary address space must have HOSTMODE=MAIN; you must start it with a startup procedure in SYS1.PROCLIB (similar to the startup procedure in Figure 13 on page 69.) If your disaster recovery policy includes aggregate backup and recovery support (ABARS), also include a second startup procedure in SYS1.PROCLIB for the DFSMShsm secondary address space.

Primary address space startup procedure

Figure 13 on page 69 is a sample DFSMShsm primary address space startup procedure.

```
//***********************************************************************/
//* SAMPLE DFSMSHSM STARTUP PROCEDURE THAT STARTS THE DFSMSHSM PRIMARY */
//* ADDRESS SPACE.                                                      */
//***********************************************************************/
//*
//DFSMSHSM  PROC CMD=00,        USE PARMLIB MEMBER ARCCMD00
//            EMERG=NO,           ALLOW ALL DFSMSHSM FUNCTIONS
//            LOGSW=YES,          SWITCH LOGS AT STARTUP
//            STARTUP=YES,        STARTUP INFO PRINTED AT STARTUP
//            UID=HSM,            DFSMSHSM-AUTHORIZED USER ID
//            SIZE=0M,            REGION SIZE FOR DFSMSHSM
//            DDD=50,             MAX DYNAMICALLY ALLOCATED DATA SETS
//            HOST=?HOST,         PROC.UNIT ID AND LEVEL FUNCTIONS
//            PRIMARY=?PRIMARY,    LEVEL FUNCTIONS
//            PDA=YES,            BEGIN PDA TRACING AT STARTUP
//            CDSR=YES            RESERVE CONTROL DATA SET VOLUMES
//DFSMSHSM  EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//          PARM=('EMERG=&EMERG','LOGSW=&LOGSW','CMD=&CMD','UID=&UID',
//          'STARTUP=&STARTUP','HOST=&HOST','PRIMARY=&PRIMARY',
            'PDA=&PDA','CDSR=&CDSR')
//******************************************************************/
//*   HSMPARM DD must be deleted from the JCL or made into a      */
//*       a comment to use Concatenated Parmlib Support           */
//******************************************************************/
//HSMPARM  DD DSN=SYS1.PARMLIB,DISP=SHR
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//SYSPRINT DD SYSOUT=A,FREE=CLOSE
//SYSUDUMP DD SYSOUT=A
//*
//******************************************************************/
//* THIS PROCEDURE ASSUMES A SINGLE CLUSTER MCDS.  IF MORE THAN   */
//* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER    */
//* CDS.                                                           */
//******************************************************************/
//*
//MIGCAT   DD DSN=HSM.MCDS,DISP=SHR
//JOURNAL  DD DSN=HSM.JRNL,DISP=SHR
//ARCLOGX  DD DSN=HSM.HSMLOGX1,DISP=OLD
//ARCLOGY  DD DSN=HSM.HSMLOGY1,DISP=OLD
//ARCPDOX  DD DSN=HSM.HSMPDOX,DISP=OLD
//ARCPDOY  DD DSN=HSM.HSMPDOY,DISP=OLD
//*
```

Figure 13. Sample Startup Procedure for the DFSMShsm Primary Address Space

Figure 14 is a sample startup procedure using STR.

```
Example of a startup procedure:
//DFSMSHSM  PROC  CMD=00,         USE PARMLIB MEMBER ARCCMD00
//                STR=00,         STARTUP PARMS IN ARCSTR00
//                HOST=?HOST,     PROC UNIT AND LEVEL FUNCTIONS
//                PRIMARY=?PRIMARY,LEVEL FUNCTIONS
//                DDD=50,         MAX DYNAMICALLY ALLOCATED DS
//                SIZE=0M         REGION SIZE FOR DFSMSHSM
//DFSMSHSM EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//         PARM=('STR=&STR','CMD=&CMD','HOST=&HOST',
                 'PRIMARY=&PRIMARY')
//HSMPARM  DD DSN=SYS1.PARMLIB,DISP=SHR
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//SYSPRINT DD SYSOUT=A,FREE=CLOSE
             . . .
PARMLIB Member ARCSTR00 contains 4 records:
 1st record: EMERG=NO,CDSQ=YES,STARTUP=YES
 2nd record:  /* This is a comment.
 3rd record:  /* This is another comment.  */
 4nd record: PDA=YES,LOGSW=YES
```

Figure 14. Sample of STR Usage

For an explanation of the keywords, see "Startup procedure keywords" on page 302.

The CMD=00 keyword refers to the ARCCMD00 member of PARMLIBs discussed in "Parameter libraries (PARMLIB)" on page 297. You can have as many ARCCMD*xx* and ARCSTR*xx* members as you need in the PARMLIBs. DFSMShsm does not require the values of CMD= and STR= to be the same, but you may want to use the same values to indicate a given configuration. In this publication, the ARCCMD member is referred to generically as ARCCMD*xx* because each different ARCCMD*xx* member can be identified by a different number.

Much of the rest of this discussion pertains to what to put into the ARCCMD*xx* member.

For information about the ARCCMD*xx* member in a multiple DFSMShsm-host environment, see "Defining all DFSMShsm hosts in a multiple-host environment" on page 255. To minimize administration, we suggest that you use a single ARCCMD*xx* and a single ARCSTR*xx* member for all DFSMShsm hosts sharing a common set of control data sets in an HSMplex.

Secondary address space startup procedure

Figure 15 is a sample DFSMShsm secondary address space startup procedure.

```
//*******************************************************************/
//* SAMPLE AGGREGATE BACKUP AND RECOVERY STARTUP PROCEDURE THAT STARTS */
//* THE ABARS SECONDARY ADDRESS SPACE.                                */
//*******************************************************************/
//*
//DFHSMABR PROC
//DFHSMABR EXEC PGM=ARCWCTL,REGION=0M
//SYSUDUMP DD SYSOUT=A
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//*
```

Figure 15. Sample Aggregate Backup and Recovery Startup Procedure

The private (24-bit) and extended private (31-bit) address space requirements for DFSMShsm are dynamic. DFSMShsm's region size should normally default to the private virtual address space (REGION=0).

To run ABARS processing, each secondary address space for aggregate backup or aggregate recovery requires 6 megabytes (MB). Three MBs of this ABARS secondary address space are above the line (in 31-bit extended private address space). The other three MBs are below the line (in 24-bit address space). An option that can directly increase this requirement is the specification of SETSYS ABARSBUFFERS(n). If this is specified with an 'n' value greater than one, use the following quick calculation to determine the approximate storage above the line you will need:

```
2MB + ('n' * 1MB)          'n' = number specified in SETSYS ABARSBUFFERS
```

As you add more functions and options to the DFSMShsm base product, the region-size requirement increases. You should therefore include the maximum region size in your setup procedure.

For a detailed discussion of the DFSMShsm primary address space startup procedure, the ABARS secondary address space startup procedure, and the startup procedure keywords, see "DFSMShsm procedures" on page 301.

Establishing the START command in the COMMNDnn member

When you initialize the MVS operating system, you want DFSMShsm to start automatically. You direct DFSMShsm to start when the MVS operating system is initialized by adding the following command to the SYS1.PARMLIB.

COM='S DFSMSHSM ***parameters*****'**

You can also start DFSMShsm from the console. DFSMShsm can be run only as a started task and never as a batch job.

DFSMShsm can run concurrently with another space-management product. This can be useful if you are switching from another product to DFSMShsm, and do not want to recall many years' worth of data just to switch to the new product over a short period like a weekend. By running the two products in parallel, you can recall data automatically from the old product, and migrate all new data with DFSMShsm.

What makes this possible is that the other product usually provides a module that must be renamed to IGG026DU to serve as the automatic locate intercept for recall. Instead, rename this module to **$IGG26DU**, and link-edit this module to the existing IGG026DU which DFSMS ships for DFSMShsm. In this manner, for each locate request, DFSMShsm's IGG026DU gives the other product control via $IGG26DU, providing it a chance to perform the recall if the data was migrated by that product. After control returns, DFSMShsm then proceeds to recall the data set if it is still migrated.

Establishing SMS-related conditions in storage groups and management classes

For your SMS-managed data sets, you must establish a DFSMShsm environment that coordinates the activities of both DFSMShsm and SMS. You can define your storage groups and management classes at one time and can modify the appropriate attributes for DFSMShsm management of data sets at another time.

The storage group contains one attribute that applies to all DFSMShsm functions, the status attribute. DFSMShsm can process volumes in storage groups having a status of ENABLE, DISNEW (disable new for new data set allocations), or QUINEW (quiesce new for new data set allocations). The other status attributes QUIALL (quiesce for all allocations), DISALL (disable all for all data set allocations), and NOTCON (not connected) prevent DFSMShsm from processing any volumes in the storage group so designated. Refer to *z/OS DFSMSdfp Storage Administration* for an explanation of the status attribute and how to define storage groups.

Writing an ACS routine that directs DFSMShsm-owned data sets to non-SMS-managed storage

Programming Interface Information

DFSMShsm must be able to direct allocation of data sets it manages to its owned storage devices so that backup versions of data sets go to backup volumes, migration copies go to migration volumes, and so forth. DFSMShsm-owned DASD volumes are not SMS-managed. If SMS were allowed to select volumes for DFSMShsm-owned data sets, DFSMShsm could not control which volumes were selected. If SMS is allowed to allocate the DFSMShsm-owned data sets to a volume other than the one selected by DFSMShsm, DFSMShsm detects that the data set is allocated to the wrong volume and fails the function being performed. Therefore, include a filter routine (similar to the sample routine in Figure 16) within your automatic class selection (ACS) routine that filters DFSMShsm-owned data sets to non-SMS managed volumes. For information on the SMS-management of DFSMShsm-owned tape volumes, see Chapter 10, "Implementing DFSMShsm tape environments," on page 189.

End Programming Interface Information

```
/***********************************************************************/
/* SAMPLE ACS ROUTINE THAT ASSIGNS A NULL STORAGE CLASS TO            */
/* DFSMSHSM-OWNED DATA SETS INDICATING THAT THE DATA SET SHOULD NOT BE */
/* SMS-MANAGED.                                                        */
/***********************************************************************/
/*                                                                     */
PROC &STORCLAS

 SET &STORCLAS = 'SCLASS2'

 FILTLIST &HSMLQ1 INCLUDE('DFHSM','HSM')
 FILTLIST &HSMLQ2 INCLUDE('HMIG','BACK','VCAT','SMALLDS','VTOC',
                          'DUMPVTOC','MDB')

 IF &DSN(1) = &HSMLQ1 AND
    &DSN(2) = &HSMLQ2 THEN
       SET &STORCLAS = '

END
/*                                                                     */
```

Figure 16. Sample ACS Routine that Directs DFSMShsm-Owned Data Sets to Non-SMS-Managed Storage

The high-level qualifiers for &HSMLQ1 and &HSMLQ2 are the prefixes that you specify with the BACKUPPREFIX (for backup and dump data set names) and the MIGRATEPREFIX (for migrated copy data set names). If you do not specify prefixes, specify the user ID from the UID parameter of the DFSMShsm startup procedure (shown in topic "Starter set example" on page 109). These prefixes and how to specify them are discussed in the *z/OS DFSMShsm Storage Administration*.

Directing DFSMShsm temporary tape data sets to tape

Programming Interface Information

It is often efficient to direct tape allocation requests to DASD when the tapes being requested are for temporary data sets. However, DFSMShsm's internal naming conventions request temporary tape allocations for backup of DFSMShsm control data sets. Therefore, it is important to direct DFSMShsm tape requests to tape.

End Programming Interface Information

If your ACS routines direct temporary data sets to DASD, DFSMShsm allocation requests for temporary tape data sets should be allowed to be directed to tape as requested (see the sample ACS routine in Figure 17 on page 73). To identify temporary tape data sets, test the &DSTYPE variable for "TEMP", and test the &PGM variable for "ARCCTL".

```
/*********************************************************************/
/* SAMPLE ACS ROUTINE THAT PREVENTS DFSMSHSM TEMPORARY (SCRATCH TAPE)  */
/* TAPE REQUESTS FROM BEING REDIRECTED TO DASD.                       */
/*********************************************************************/

                  :
                  :
                  :

/*********************************************************************/
/*                SET FILTLIST FOR PRODUCTION DATA SETS               */
/*********************************************************************/

 FILTLIST EXPGMGRP INCLUDE('ARCCTL')

                  :
                  :
                  :

/*********************************************************************/
/* FILTER TEMPORARY (SCRATCH TAPE) TAPE REQUESTS INTO DFSMSHSM         */
/* REQUESTS AND NON-DFSMSHSM REQUESTS.  SEND DFSMSHSM REQUESTS TO TAPE */
/* AS REQUESTED.  SEND NON-DFSMSHSM REQUESTS TO DASD.                  */
/*********************************************************************/

IF (&DSTYPA = 'TEMP' && &UNIT = &TAPE_UNITS && &PGM ^= &EXPGMGRP)

   THEN DO
          SET &STORCLAS = 'BASE'
          WRITE '***************************************************'
          WRITE '* THIS TAPE DATA SET HAS BEEN REDIRECTED TO TAPE   *'
          WRITE '***************************************************'
       END
```

Figure 17. Sample ACS Routine That Prevents DFSMShsm Temporary Tape Requests from being Redirected to DASD

Establishing the ARCCMDxx member of a PARMLIB

At DFSMShsm startup, DFSMShsm reads the ARCCMD*xx* parameter library (PARMLIB) member that is pointed to by the DFSMShsm startup procedure or is found in the MVS concatenated PARMLIB data sets.

An ARCCMD*xx* member consisting of DFSMShsm commands that define your site’s DFSMShsm processing environment must exist in a PARMLIB data set. (The PARMLIB containing the ARCCMD*xx* member may be defined in the startup procedure.) An example of the ARCCMD*xx* member can be seen starting at “Starter set example” on page 109.

Modifying the ARCCMDxx member

In most cases, adding a command to the ARCCMD*xx* member provides an addendum to any similar command that already exists in the member. For example, the ARCCMD*xx* member that exists from the starter set contains a set of commands with their parameters. You can remove commands that do not meet your needs from the ARCCMD*xx* member and replace them with commands that do meet your needs.

ARCCMDxx member for the starter set

The ARCCMD*xx* member provided with the starter set is written to accommodate any system so some commands are intentionally allowed to default and others specify parameters that are not necessarily optimal. Because the starter set does not provide an explanation of parameter options, we discuss the implications of choosing SETSYS parameters in this section.

Issuing DFSMShsm commands

DFSMShsm commands can be issued from the operator's console, from a TSO terminal, as a CLIST from a TSO terminal, as a job (when properly surrounded by JCL) from the batch reader, or from a PARMLIB member. DFSMShsm commands can be up to 1024 bytes long. The *z/OS DFSMShsm Storage Administration* explains how to issue the DFSMShsm commands and why to issue them.

Implementing new DFSMShsm ARCCMDxx functions

If you have DFSMShsm running with an established ARCCMD*xx* member, for example ARCCMD00, you can copy the ARCCMD*xx* member to a member with another name, for example, ARCCMD01. You can then modify the new ARCCMD*xx* member by adding and deleting parameters.

To determine how the new parameters affect DFSMShsm's automatic processes, run DFSMShsm in DEBUG mode with the new ARCCMD*xx* member. See "Debug mode of operation for gradual conversion to DFSMShsm" on page 378 and the *z/OS DFSMShsm Storage Administration* for an explanation of running DFSMShsm in DEBUG mode.

Defining storage administrators to DFSMShsm

As part of defining your DFSMShsm environment, you must designate storage administrators and define their authority to issue authorized DFSMShsm commands. The authority to issue authorized commands is granted either through RACF FACILITY class profiles or the DFSMShsm AUTH command.

Because DFSMShsm operates as an MVS-authorized task, it can manage data sets automatically, regardless of their security protection. DFSMShsm allows an installation to control the authorization of its commands through the use of either RACF FACILITY class profiles or the AUTH command.

If the RACF FACILITY class is active, DFSMShsm always uses it to protect all DFSMShsm commands. If the RACF FACILITY class is not active, DFSMShsm uses the AUTH command to protect storage administrator DFSMShsm commands. There is no protection of user commands in this environment.

The RACF FACILITY class environment

DFSMShsm provides a way to protect all DFSMShsm command access through the use of RACF FACILITY class profiles. An active RACF FACILITY class establishes the security environment.

An individual, such as a security administrator, defines RACF FACILITY class profiles to grant or deny permission to issue individual DFSMShsm commands.

For more information about establishing the RACF FACILITY class environment, see "Authorizing and protecting DFSMShsm commands in the RACF FACILITY class environment" on page 173.

The DFSMShsm AUTH command environment

If you are not using the RACF FACILITY class to protect all DFSMShsm commands, the AUTH command is used to protect DFSMShsm-authorized commands.

To prevent unwanted changes to the parameters that control all data sets, commands within DFSMShsm are classified as *authorized* and *nonauthorized*.

Authorized commands can be issued only by a user specifically authorized by a storage administrator. Generally, authorized commands can affect data sets not owned by the person issuing the command and should, therefore, be limited to only those whom you want to have that level of control.

Nonauthorized commands can be issued by any user, but they generally affect only those data sets for which the user has appropriate security access. Nonauthorized commands are usually issued by system users who want to manage their own data sets with DFSMShsm user commands.

DFSMShsm has two categories of authorization: USER and CONTROL.

If you specify . . .	Then . . .
AUTH U012345 DATABASEAUTHORITY(USER)	User U012345 can issue any DFSMShsm command except the command that authorizes other users.
AUTH U012345 DATABASEAUTHORITY(CONTROL)	DFSMShsm gives user U012345 authority to issue the AUTH command to authorize other users. User U012345 can then issue the AUTH command with the DATABASEAUTHORITY(USER) parameter to authorize other storage administrators who can issue authorized commands.

Anyone can issue authorized commands from the system console, but they cannot authorize other users. The ARCCMD*xx* member *must* contain an AUTH command granting CONTROL authority to a storage administrator. That storage administrator can then authorize or revoke the authority of other users as necessary. If no AUTH command grants CONTROL authority to any user, no storage administrator can authorize any other user. If the ARCCMD*xx* member does not contain *any* AUTH command, authorized commands can be issued only at the operator’s console.

Defining the DFSMShsm MVS environment

You define the MVS environment to DFSMShsm when you specify:

- The job entry subsystem
- The amount of common service area storage
- The sizes of cell pools
- Operator intervention in DFSMShsm automatic operation
- Data set serialization
- Swap capability of DFSMShsm’s address space
- Maximum secondary address space

Each of the preceding tasks relates to a SETSYS command in the ARCCMD*xx* member.

Figure 18 on page 76 is an example of the commands that define an MVS environment:

```
 /***********************************************************************/
/*  SAMPLE SETSYS COMMANDS THAT DEFINE THE DEFAULT MVS ENVIRONMENT     */
/***********************************************************************/
/*
SETSYS JES2
SETSYS CSALIMITS(MAXIMUM(100) ACTIVE(90) INACTIVE(30) MWE(4))
SETSYS NOREQUEST
SETSYS USERDATASETSERIALIZATION
SETSYS NOSWAP
SETSYS MAXABARSADDRESSSPACE(1)
/*
```

Figure 18. Sample SETSYS Commands That Define the Default MVS Environment

Specifying the job entry subsystem

As part of defining your MVS environment to DFSMShsm, you must identify the job entry subsystem (JES) at your site as either JES2 or JES3 by specifying the SETSYS(JES2|JES3) command in the ARCCMD*xx* member. The ARCCMD*xx* is located in a PARMLIB.

JES3 considerations

When you implement DFSMShsm in a JES3 environment, you must observe certain practices and restrictions to ensure correct operation:

- For a period of time after the initialization of JES3 and before the initialization of DFSMShsm, all JES3 locates will fail. To reduce this exposure:
  - Start DFSMShsm as early as possible after the initialization of JES3.
  - Specify the SETSYS JES3 command as early as possible in the startup procedure and before any ADDVOL commands.
- Specify JES3 during DFSMShsm startup when DFSMShsm is started in a JES3 system. This avoids an error message being written when DFSMShsm receives the first locate request from the JES3 converter/interpreter.
- Depend on the computing system catalog to determine the locations of data sets.
- Do not allocate the control data sets and the JES3 spool data set on the same volume because you could prevent DFSMShsm from starting on a JES3 local processor.
- All devices that contain volumes automatically managed or processed by DFSMShsm must be controlled by JES3. All volumes managed by DFSMShsm (even those managed by command) should be used on devices controlled by JES3.
- DFSMShsm must be active on the processing units that use volumes managed by DFSMShsm and on any processing unit where JES3 can issue the locate request for the setup of jobs that use volumes managed by DFSMShsm.

The specification of JES3 places a constraint on issuing certain DFSMShsm commands. When you use JES2, you can issue ADDVOL, DEFINE, and SETSYS commands at any time. When you specify JES3, you must issue ADDVOL commands for primary volumes, DEFINE commands for pools (except aggregate recovery pools), and the SETSYS JES2 or SETSYS JES3 commands in the ARCCMD*xx* member. In addition, if you are naming tape devices with esoteric names, you must include the SETSYS USERUNITTABLE command in the ARCCMD*xx* member before the ADDVOL command for any of the tapes that are in groups defined with esoteric names.

If you specify JES3 but the operating system uses JES2, DFSMShsm is not notified of the error. However, DFSMShsm uses the rules that govern pool configuration for JES3, and one or both of the following situations can occur:

- Some ADDVOL, SETSYS, and DEFINE commands fail if they are issued when they are not acceptable in a JES3 system.
- Volumes eligible for recall in a JES2 system might not qualify for the DFSMShsm general pool and, in some cases, are not available for recall in the JES3 system.

When you use DFSMShsm and JES3, the usual configuration is a *symmetric* configuration. A symmetric configuration is one where the primary volumes are added to DFSMShsm in all processing units and the hardware is connected in all processing units. Because of the dynamic reconfiguration of JES3, you should use a symmetric JES3 configuration.

If your device types are 3490, define the special esoteric names SYS3480R and SYS348XR to JES3. This may only be done after the system software support (JES3, DFP, and MVS) for 3490 is available on all processing units.

The main reason for this is conversion from 3480s, to allow DFSMShsm to convert the following generic unit names to the special esoteric names:

- 3480 (used for output) is changed to SYS3480R for input drive selection. SYS3480R is a special esoteric name that is associated with all 3480, 3480X, and 3490 devices. Any device in this esoteric is capable of reading a cartridge written by a 3480 device.
- 3480X (used for output) is changed to SYS348XR for input drive selection. SYS348XR is a special esoteric name that is associated with all 3480X and 3490 devices. Any device in this esoteric is capable of reading a cartridge written by a 3480X device.

Note:

1. Because of the DFSMShsm use of the S99DYNDI field in the SVC99 parameter list, the JES3 exit IATUX32 is not invoked when DFSMShsm is active.
2. By default, JES3 support is not enabled for DFSMShsm hosts defined using HOSTMODE=AUX. Contact IBM support if you require JES3 support for AUX DFSMShsm hosts. When JES3 for AUX DFSMShsm hosts is enabled, you should start the main DFSMShsm host before starting any AUX hosts and stop all AUX hosts before stopping the main host.

Specifying the amount of common service area storage

Common Service Area (CSA) storage is cross-memory storage (accessible to any address space in the system) for management work elements (MWEs). The SETSYS CSALIMITS command determines the amount of common service area (CSA) storage that DFSMShsm is allowed for its management work elements. The subparameters of the CSALIMITS parameter specify how CSA is divided among the MWEs issued to DFSMShsm. Unless almost all of DFSMShsm's workload is initiated from an external source, the defaults are satisfactory. Figure 18 on page 76 specifies the same values as the defaults.

One MWE is generated for each request for service that is issued to DFSMShsm. Requests for service that generate MWEs include:

- Batch jobs that need migrated data sets
- Both authorized and nonauthorized DFSMShsm commands including TSO requests to migrate, recall, and back up data sets

Two types of MWEs can be issued: wait and nowait. A WAIT MWE remains in CSA until DFSMShsm finishes acting on the request. A NOWAIT MWE remains in CSA under control of the MWE subparameter until DFSMShsm accepts it for processing. The NOWAIT MWE is then purged from CSA unless the MWE subparameter of CSALIMITS specifies that some number of NOWAIT MWEs are to be retained in CSA.

Note: If you are running more than one DFSMShsm host in a z/OS image, the CSALIMITS values used are those associated with the host with HOSTMODE=MAIN. Any CSALIMITS values specified for an AUX host are ignored.

Selecting values for the SETSYS CSA command subparameters

DFSMShsm can control the amount of common-service-area (CSA) storage for management work elements (MWEs) whether or not DFSMShsm has been active during the current system initial program load (IPL). When DFSMShsm has not been active during the current IPL, DFSMShsm defaults control the amount of CSA. When DFSMShsm has been active, either the DFSMShsm defaults or SETSYS values control the amount of CSA. The DFSMShsm defaults for CSA are shown in Figure 18 on page 76. The subparameters of the SETSYS CSA command are discussed in the following:

Selecting the value for the MAXIMUM subparameter: The MAXIMUM subparameter determines the upper limit of CSA storage for cross-memory communication of MWEs. After this amount of CSA has been used, additional MWEs cannot be stored. The average MWE is 400 bytes. The DFSMShsm default for this subparameter is 100KB (1KB equals 1024 bytes).

Limiting CSA has two potential uses in most data centers; protecting other application systems from excessive CSA use by DFSMShsm or serving as an early-warning sign of a DFSMShsm problem.

Setting CSALIMIT to protect other applications: Setting CSALIMITs to protect other applications depends on the amount of CSA available in the "steady-state" condition when you know the amount of CSA left over after the other application is active. This method measures the CSA usage of applications other than DFSMShsm.

1. Run the system without DFSMShsm active.
2. Issue the QUERY CSALIMIT command to determine DFSMShsm's CSA use.
3. Set the MAXIMUM CSA subparameter to a value less than the "steady-state" amount available for the CSA.
4. Think of DFSMShsm as a critical application with high availability requirements to set the remaining CSALIMITs.

Setting CSALIMIT as an early warning: Setting CSALIMITs as an early warning is different. Rather than measuring the CSA usage of some other application, you measure DFSMShsm's CSA use. This method uses DFSMShsm CSALIMITS as an alarm system that notifies the console operator if DFSMShsm's CSA usage deviates from normal.

1. Run the system for a week or two with CSALIMIT inactive or set to a very high value.
2. Issue the QUERY CSALIMIT command periodically to determine DFSMShsm's CSA use.
3. Identify peak periods of CSA use.

4. Select a maximum value based on the peak, multiplied by a safety margin that is within the constraints of normally available CSA.

Selecting the value for the ACTIVE subparameter: The ACTIVE subparameter specifies the percentage of maximum CSA available to DFSMShsm for both WAIT and NOWAIT MWEs when DFSMShsm is active. Until this limit is reached, all MWEs are accepted. After this limit has been reached, only WAIT MWEs from batch jobs are accepted. The active limit is a percentage of the DFSMShsm maximum limit; the DFSMShsm default is 90%.

Selecting the value for the INACTIVE subparameter: The INACTIVE subparameter specifies the percentage of CSA that is available to DFSMShsm for NOWAIT MWEs when DFSMShsm is inactive. This prevents the CSA from being filled with NOWAIT MWEs when DFSMShsm is inactive.

Both the ACTIVE and INACTIVE CSALIMITs are expressed as percentages of the maximum amount of CSA DFSMShsm is limited to. Both specifications (ACTIVE and INACTIVE) affect the management of NOWAIT MWEs, which are ordinarily a small part of the total DFSMShsm workload.

The DFSMShsm default is 30%. When you start DFSMShsm, this limit changes to the active limit.

Selecting the value for the MWE subparameter: The MWE subparameter specifies the number of NOWAIT MWEs from each user address space that are kept in CSA until they are completed.

The MWE subparameter can be set to 0 if DFSMShsm is solely responsible for making storage management decisions. The benefit of setting the MWE subparameter to zero (the default is four) is that the CSA an MWE consumes is freed immediately after the MWE has been copied into DFSMShsm 's address space, making room for additional MWEs in CSA. Furthermore, if DFSMShsm is solely responsible for storage management decisions, the loss of one or more NOWAIT MWEs (such as, a migration copy that is not being deleted) when DFSMShsm is stopped could be viewed as insignificant.

The benefit of setting the MWE subparameter to a nonzero quantity is that MWEs remain in CSA until the function completes, so if DFSMShsm stops, the function is continued after DFSMShsm is restarted. The default value of 4 is sufficient to restart almost all requests; however, a larger value provides for situations where users issue many commands. MWEs are not retained across system outages; therefore, this parameter is valuable only in situations where DFSMShsm is stopped and restarted.

Restartable MWEs are valuable when a source external to DFSMShsm is generating critical work that would be lost if DFSMShsm failed. Under such conditions, an installation would want those MWEs retained in CSA until they had completed.

The decision for the storage administrator is whether to retain NOWAIT MWEs in CSA. No method exists to selectively discriminate between MWEs that should be retained and other MWEs unworthy of being held in CSA. Figure 19 on page 80 shows the three storage limits in the common service area storage.

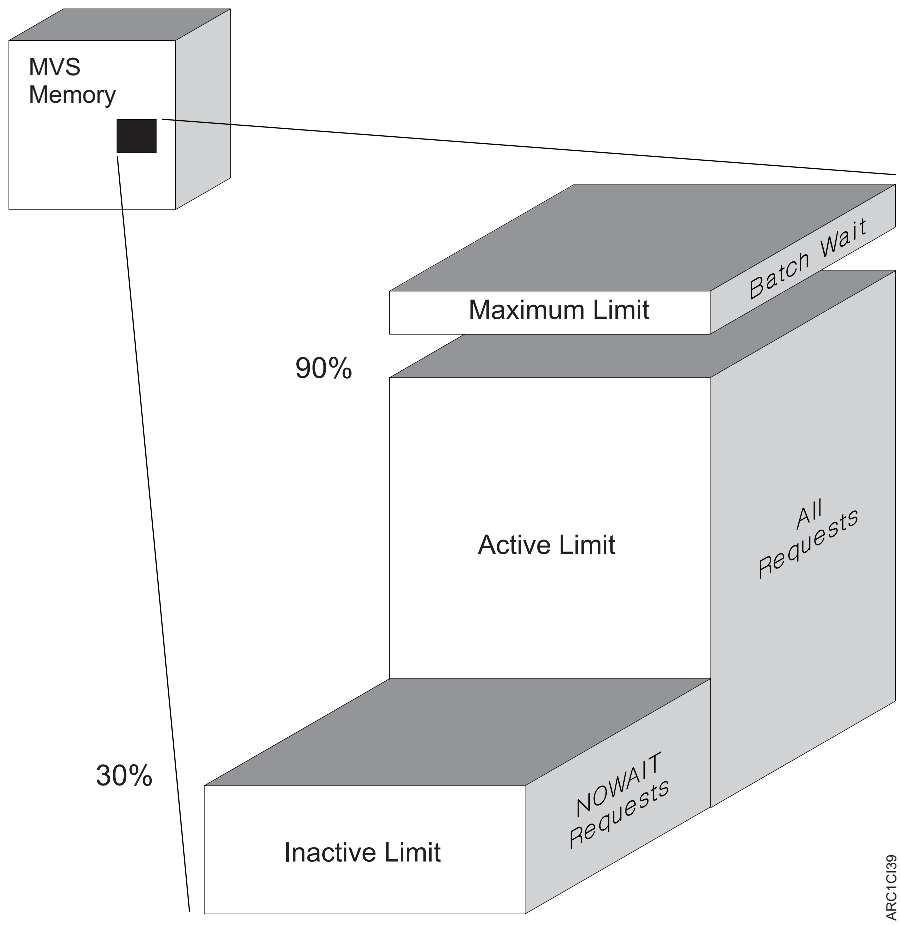


Figure 19. Overview of Common Service Area Storage

WAIT and NOWAIT MWE considerations: DFSMShsm keeps up to four NOWAIT MWEs on the CSA queue for each address space. Subsequent MWEs from the same address space are deleted from CSA when the MWE is copied to the DFSMShsm address space. When the number of MWEs per address space falls under four, MWEs are again kept in CSA until the maximum of four is reached. Table 6 shows the types of requests and how the different limits affect these requests.

Table 6. How Common Service Area Storage Limits Affect WAIT and NOWAIT Requests

Type of Request	DFSMShsm Active	DFSMShsm Inactive
Batch WAIT	If the current CSA storage is less than the maximum limit, the MWE is added to the queue. Otherwise, a message is issued and the request fails.	If the current CSA storage is less than the maximum limit, the operator is required to either start DFSMShsm or cancel the request.
TSO WAIT	If the current CSA storage is less than the active limit, the MWE is added to the queue. Otherwise, a message is issued and the request fails.	The operator is prompted to start DFSMShsm but the request fails.
NOWAIT	If the current CSA storage is less than the active limit, the MWE is added to the queue. Otherwise, a message is issued and the request fails.	If the current CSA storage is less than the inactive limit, the MWE is added to the queue. Otherwise, a message is issued and the request fails.

A system programmer can use the SETSYS command to change any one of these values. The SETSYS command is described in *z/OS DFSMShsm Storage Administration*.

Specifying the size of cell pools

DFSMShsm uses cell pools (the MVS CPOOL function) to allocate virtual storage for frequently used modules and control blocks. Cell pool storage used for control blocks is extendable, while cell pool storage used by modules is not. Using cell pools reduces DFSMShsm CPU usage and improves DFSMShsm performance. The DFSMShsm startup procedure specifies the size (in number of cells) of five cell pools used by DFSMShsm.

DFSMShsm is configured with a default size for each cell pool. You can change these sizes by changing the CELLS keyword in the startup procedure for the DFSMShsm primary address space. Typically the default values are acceptable. However, if you run many concurrent DFSMShsm tasks, you may receive an ARC0019I message, which identifies a cell pool that has run out of cells. If you receive this message, you should increase the size of the indicated cell pool by at least the number of cells specified in the message.

Related reading

- "Adjusting the size of cell pools" on page 295
- "DFSMShsm startup procedure" on page 301
- "CELLS (default = (200,100,100,50,20))" on page 305

Specifying operator intervention in DFSMShsm automatic operations

The SETSYS(REQUEST|NOREQUEST) command determines whether DFSMShsm prompts the operator before beginning its automatic functions.

If you specify . . .	Then . . .
SETSYS NOREQUEST	DFSMShsm begins its automatic functions without asking the operator.
SETSYS REQUEST	DFSMShsm prompts the operator for permission to begin its automatic functions by issuing message ARC0505D. You can write code for the MVS message exit IEAVMXIT to respond to the ARC0505D message automatically. The code could query the status of various other jobs in the system and make a decision to start or not to start the DFSMShsm automatic function, based on the workload in the system at the time.

Specifying data set serialization

When DFSMShsm is backing up or migrating data sets, it must prevent those data sets from being changed. It does this by serialization. Serialization is the process of controlling access to a resource to protect the integrity of the resource. DFSMShsm serialization is determined by the SETSYS DFHSMDATASETSERIALIZATION | USERDATASETSERIALIZATION command.

Note: In DFSMS/MVS Version 1 Release 5, the incremental backup function has been restructured in order to improve the performance of that function. The SETSYS DFHSMDATASETSERIALIZATION command disables that improvement.

Only use the SETSYS DFHSMDATASETSERIALIZATION command if your environment requires it. Otherwise, it is recommended that you use the SETSYS USERDATASETSERIALIZATION command.

If you specify . . .	Then . . .
SETSYS DFHSMDATASETSERIALIZATION	DFSMShsm issues a RESERVE command that prevents other processing units from accessing the volume while DFSMShsm is copying a data set during volume migration. To prevent system interlock, DFSMShsm releases the reserve on the volume to update the control data sets and the catalog. After the control data sets have been updated, DFSMShsm reads the data set VTOC entry for the data set that was migrated to ensure that no other processing unit has updated the data set while the control data sets were being updated. If the data set has not been updated, it is scratched. If the data set has been updated, DFSMShsm scratches the migration copy of the data set and again updates the control data sets and the catalog to reflect the current location of the data set. Multivolume non-VSAM data sets are not supported by this serialization option because of possible deadlock situations. For more information about volume reserve serialization, see "DFHSMDATASETSERIALIZATION" on page 265.
SETSYS USERDATASETSERIALIZATION	Serialization is maintained throughout the complete migration operation, including the scratch of the copy on the user volume. No other processing unit can update the data set while DFSMShsm is performing its operations, and no second read of the data set VTOC entry is required for checking. Also since there is no volume reserved while copying the data set, other data sets on the volume are accessible to users. Therefore, USERDATASETSERIALIZATION provides a performance advantage to DFSMShsm and users in those systems equipped to use it.

You may use SETSYS USERDATASETSERIALIZATION if:

- The data sets being processed are only accessible to a single z/OS image, even if you are running multiple DFSMShsm hosts in that single z/OS image.

  OR
- The data sets can be accessed from multiple z/OS images, and a product like GRS must be active and is required in a multiple-image environment.

Specifying the swap capability of the DFSMShsm address space

The SETSYS SWAP|NOSWAP command determines whether the DFSMShsm address space can be swapped out of real storage.

If you specify . . .	Then . . .
SETSYS SWAP	The DFSMShsm address space can be swapped out of real storage.

If you specify . . .	Then . . .
SETSYS NOSWAP	The DFSMShsm address space cannot be swapped out of real storage.

Guideline: The NOSWAP option is recommended. DFSMShsm always sets the option to NOSWAP when the ABARS secondary address space is active.

In a multisystem environment, DFSMShsm also always sets the option to NOSWAP so that cross-system coupling facility (XCF) functions are available. See Chapter 12, “DFSMShsm in a sysplex environment,” on page 279 for the definition of a multisystem (or a sysplex) environment.

Specifying maximum secondary address space

The SETSYS MAXABARSADDRESSSPACE (*number*) command specifies the maximum number of aggregate backup and recovery secondary address spaces that DFSMShsm allows to process concurrently. The SETSYS ABARSPROCNAME(*name*) command specifies the name of the procedure that starts an ABARS secondary address space.

Defining the DFSMShsm security environment for DFSMShsm-owned data sets

The SETSYS commands control the relationship of DFSMShsm to RACF and control the way DFSMShsm prevents unauthorized access to DFSMShsm-owned data sets. You can use the following SETSYS commands to define your security environment:

- How DFSMShsm determines the user ID when RACF is not installed and active.
- Whether to indicate that migration copies and backup versions of data sets are RACF protected.
- How DFSMShsm protects scratched data sets.

Figure 20 is an example of a typical DFSMShsm security environment.

```
/***********************************************************************/
/* SAMPLE SETSYS COMMANDS THAT DEFINE THE DFSMSHSM SECURITY ENVIRONMENT*/
/***********************************************************************/
/*
SETSYS NOACCEPTPSCBUSERID
SETSYS NOERASEONSCRATCH
SETSYS NORACFIND
/*
```

Figure 20. Sample SETSYS Commands to Define the Security Environment for DFSMShsm

DFSMShsm maintains the security of those data sets that are RACF protected.

DFSMShsm does not check data set security for:

- Automatic volume space management
- Automatic dump
- Automatic backup
- Automatic recall
- Operator commands entered at the system console

- Commands issued by a DFSMShsm-authorized user

DFSMShsm checks security for data sets when a user who is not DFSMShsm-authorized issues a nonauthorized user command (HALTERDS, HBDELETE, HMIGRATE, HDELETE, HBACKDS, HRECALL, or HRECOVER). Security checking is not done when DFSMShsm-authorized users issue the DFSMShsm user commands. If users are not authorized to manipulate data, DFSMShsm does not permit them to alter the backup parameters of a data set, delete backup versions, migrate data, delete migrated data, make backup versions of data, recall data sets, or recover data sets.

Authorization checking is done for the HCANCEL and CANCEL commands. However the checking does not include security checking the user's authority to access a data set. Whether a user has comprehensive or restricted command authority controls whether RACF authority checking is performed for each data set processed by the ABACKUP command. Refer to *z/OS DFSMShsm Storage Administration* for more information about authorization checking during aggregate backup.

Determining batch TSO user IDs

When a TSO batch job issues a DFSMShsm-authorized command, DFSMShsm must be able to verify the authority of the TSO user ID to issue the command. For authorization checking purposes when processing batch TSO requests, DFSMShsm obtains a user ID as follows:

- If RACF is active, the user ID is taken from the access control environment element (ACEE), a RACF control block.
- If RACF is not active and the SETSYS ACCEPTPSCBUSERID command has been specified, the user ID is taken from the TSO-protected step control block (PSCB). If no user ID is present in the PSCB, the user ID is set to **BATCH*. It is the installation's responsibility to ensure that a valid user ID is present in the PSCB.
- If RACF is not active and the SETSYS ACCEPTPSCBUSERID command has not been specified or if the default (NOACCEPTPSCBUSERID) has been specified, the user ID is set to **BATCH* for authorization checking purposes.

If you have RACF installed and active, you can specify that RACF protect resources; therefore, specify NOACCEPTPSCBUSERID. (NOACCEPTPSCBUSERID has no relevance but is included for completeness. However, if your system does not have RACF installed and active, you should use ACCEPTPSCBUSERID.)

The NOACCEPTPSCBUSERID parameter specifies how DFSMShsm determines the user ID for TSO submission of DFSMShsm-authorized commands in systems that do not have RACF installed and active.

Specifying whether to indicate RACF protection of migration copies and backup versions of data sets

When DFSMShsm migrates or backs up a data set, it can indicate that the copy is protected by a RACF discrete profile. Such a data set, when its indicator is on, is called RACF-indicated. RACF indication provides protection only for data sets that are RACF-indicated on the level 0 volume and it allows only the RACF security administrator to directly access the migration and backup copies.

For a non-VSAM data set, the RACF indicator is a bit in the volume table of contents (VTOC) of the DASD volume on which the data set resides.

For a VSAM data set, the RACF indicator is a bit in the catalog record. The indicator remains with the data set even if the data set is moved to another system. However, if the data set profile fails to move or is somehow lost, a RACF security administrator must take action before anyone can access the data set.

The SETSYS RACFIND|NORACFIND command determines whether DFSMShsm-owned data sets are RACF-indicated.

If you specify . . .	Then . . .
SETSYS RACFIND	DFSMShsm sets the RACF indicator in the data set VTOC entry for migration copies and backup versions. The RACFIND option is recommended for systems that do not have an always-call environment, do not have generic profiles enabled, but do have RACF discrete data set profiles.
SETSYS NORACFIND	DFSMShsm does not perform I/O operations to turn on the RACF indicator for migration copies and backup versions when RACF-indicated data sets are migrated and backed up to DASD.

Before specifying the SETSYS NORACFIND command, ensure that you:
- Define a generic profile for the prefixes of DFSMShsm-owned data sets
- Enable generic DATASET profiles

The preferred implementation is to create an environment in which you can specify the NORACFIND option. Generic profiles enhance DFSMShsm performance because DFSMShsm does not perform I/O operations to turn on the RACF-indicated bit.

For a discussion of RACF environments and profiles, refer to *z/OS DFSMShsm Storage Administration*.

Specifying security for scratched DFSMShsm-owned DASD data sets

Some data sets are so sensitive that you must ensure that DASD residual data cannot be accessed after they have been scratched. RACF has a feature to erase the space occupied by a data set when the data set is scratched from a DASD device. This feature, called erase-on-scratch, causes overwriting of the DASD residual data by data management when a data set is deleted.

If you specify . . .	Then . . .
SETSYS ERASEONSCRATCH	Erase-on-scratch processing is requested only for DFSMShsm-owned DASD data sets.

When the ERASEONSCRATCH parameter is in effect, DFSMShsm queries RACF for the erase status of the user’s data set for use with the backup version or the migration copy. If the erase status from the RACF profile is ERASE when the backup version or the migration copy is scratched, the DASD residual data is overwritten by data management. If the erase status from the RACF profile is NOERASE when the backup version or the migration copy is scratched, the DASD residual data is not overwritten by data management.

The ERASEONSCRATCH parameter has no effect on data sets on level 0 volumes on which the RACF erase attribute is supported. The ERASEONSCRATCH parameter allows the erase attribute to be carried over to migration copies and backup versions.

Note: Records making up a data set in a small-data-set-packing (SDSP) data set are not erased. Refer to *z/OS DFSMShsm Storage Administration* for information about small-data-set-packing data set security.

If you specify . . .	Then . . .
SETSYS NOERASEONSCRATCH	No erase-on-scratch processing is requested for DFSMShsm-owned volumes.

Erase-on-scratch considerations

Before you specify the erase-on-scratch option for integrated catalog facility (ICF) cataloged VSAM data sets that have the ERASE attribute and have backup profiles, consider the following results:

- DFSMShsm copies of ICF cataloged VSAM data sets with the ERASE attribute indicated in the *RACF profile* are erased with the same erase-on-scratch support as for all other data sets.

  DFSMShsm *does not* migrate ICF cataloged VSAM data sets that have the ERASE attribute in the *catalog record*. The migration fails with a return code 99 and a reason code 2 indicating that the user can remove the ERASE attribute from the catalog record and can specify the attribute in the RACF profile to obtain DFSMShsm migration and erase-on-scratch support of the data set.
- ERASE status is obtained only from the original RACF profile. Backup profiles created by DFSMShsm (refer to *z/OS DFSMShsm Storage Administration*) are not checked. The original ERASE attribute is saved in the backup version (C) record at the time of backup and is checked at recovery time if the original RACF profile is missing.
- The records in an SDSP data set are not overwritten on recall even if the SETSYS ERASEONSCRATCH command has been specified. When a data set is recalled from an SDSP data set, the records are read from the control interval and returned as a data set to the level 0 volume. When migration cleanup is next performed, the VSAM erase process reformats the control interval but does not overwrite any residual data. Erase-on-scratch is effective for SDSP data sets only when the SDSP data set itself is scratched. Refer to *z/OS DFSMShsm Storage Administration* for a discussion of protecting small-data-set-packing data sets.

Defining data formats for DFSMShsm operations

Because DFSMShsm moves data between different device types with different device geometries, the format of data can change as it moves from one device to another.

There are three data formats for DFSMShsm operations:

- The format of the data on DFSMShsm-owned volumes
- The blocking of the data on DFSMShsm-owned DASD volumes
- The blocking of data sets that are recalled and recovered

You can control each of these format options by using SETSYS command parameters. The parameters control the data compaction option, the optimum DASD blocking option (see "Optimum DASD blocking option" on page 90), the use of the tape device improved data recording capability, and the conversion option. You can also use DFSMSdss dump COMPRESS for improved tape utilization. Refer to *z/OS DFSMShsm Storage Administration* for additional information about invoking full-volume dump compression. Figure 21 lists sample SETSYS commands for defining data formats.

```
/***********************************************************************/
/*               SAMPLE DFSMSHSM DATA FORMAT DEFINITIONS               */
/***********************************************************************/
/*
SETSYS COMPACT(DASDMIGRATE NOTAPEMIGRATE DASDBACKUP NOTAPEBACKUP)
SETSYS COMPACTPERCENT(30)
SETSYS OBJECTNAMES(OBJECT,LINKLIB)
SETSYS SOURCENAMES(ASM,PROJECT)
SETSYS OPTIMUMDASDBLOCKING
SETSYS CONVERSION(REBLOCKTOANY)
SETSYS TAPEHARDWARECOMPACT
/*
```

Figure 21. Sample Data Format Definitions for a Typical DFSMShsm Environment

Data compaction option

The data compaction option can save space on migration and backup volumes by encoding each block of each data set that DFSMShsm migrates or backs up. DFSMShsm compacts data with the Huffman Frequency Encoding compaction algorithm. The compacted output blocks can vary in size. An input block consisting of many least-used EBCDIC characters can be even longer after being encoded. If this occurs, DFSMShsm passes the original data block without compaction to the output routine.

Whether DFSMShsm compacts each block of data as the data is backed up or migrated from a level 0 volume is determined by the SETSYS COMPACT command. DFSMShsm compacts each block of data as the data set is backed up or migrated from a level 0 volume. Compaction or decompaction never occurs when a data set moves from one migration volume to another or from one backup volume to another. DFSMShsm does not compact data sets when they are migrated for extent reduction, are in compressed format, or during DASD conversion.

If you specify . . .	Then . . .
SETSYS COMPACT(DASDMIGRATE NOTAPEMIGRATE DASDBACKUP NOTAPEBACKUP)	Every block of data that migrates or is backed up to DASD is a candidate for compaction.

When DFSMShsm recalls or recovers a compacted data set, DFSMShsm automatically decodes and expands the data set. DFSMShsm decompacts encoded data even if you later run with SETSYS COMPACT(NONE).

If you do not want a specific data set to be compacted during volume migration or volume backup, invoke the data set migration exit (ARCMDEXT) or the data set backup exit (ARCBDEXT) to prevent compaction of that data set. For more information about the data set migration exit and the data set backup exit, refer to *z/OS DFSMS Installation Exits*.

Compaction tables

When choosing an algorithm for compacting a data set, DFSMShsm selects either the unique source or object compaction table or selects the default general compaction table. You can identify data sets that you want to compact with unique source or object compaction tables by specifying the low-level qualifiers for those data sets when you specify the SETSYS SOURCENAMES and SETSYS OBJECTNAMES commands.

For generation data groups, DFSMShsm uses the next-to-the-last qualifier of the data set name. DFSMShsm uses the same compaction table for all blocks in each data set. The source compaction table is designed to compact data sets that contain programming language source code. The object compaction table is designed to compact data sets containing object code and is based on an expected frequency distribution of byte values.

Compaction percentage

When compacting a data set during migration or backup, DFSMShsm keeps a running total of the number of bytes in each compacted block that is written to the migration or backup volume. DFSMShsm also keeps a running total of the number of bytes that were in the blocks before compaction. With these values, DFSMShsm determines the space savings value, expressed as a percentage.

$$\text{Space Savings} = \frac{\text{Total Bytes Before Compaction} - \text{Total Bytes After Compaction}}{\text{Total Bytes Before Compaction}} \times 100$$

DFSMShsm uses the space savings percentage to determine if it should compact recalled or recovered data sets when they are subsequently backed up or migrated again. You specify this space saving percentage when you specify the SETSYS COMPACTPERCENT command.

At least one track on the DASD migration or backup volume must be saved, or the compacted data set is not eligible for compaction when it is subsequently migrated or backed up.

Note: For SDSP data sets, DFSMShsm considers only the space saving percentage because small-data-set packing is intended for small user data sets where the space savings is typically less than a track.

If you specify . . .	Then . . .
SETSYS COMPACT (DASDMIGRATE | TAPEMIGRATE)	DFSMShsm compacts each record of a data set on a level 0 volume the first time it migrates the data set. During subsequent migrations from level 0 volumes (as a result of recall), DFSMShsm performs additional compaction of the data set only if the percentage of space savings (as indicated from the original backup or migration) exceeds the value specified with the SETSYS COMPACTPERCENT command.
SETSYS COMPACT (DASDBACKUP | TAPEBACKUP)	DFSMShsm compacts each record of a data set on a level 0 volume the first time it backs up the data set. During subsequent backups (as a result of recovery), DFSMShsm performs additional compaction of the data set only if the percentage of space savings (as indicated by the original backup) exceeds the value specified with the SETSYS COMPACTPERCENT command.

DFSMShsm stores the space saving percentage in the MCDS data set (MCD record) or the BCDS data set (MCB record). If the MCD or MCB record is deleted (for example, during migration cleanup or expiration of backup versions), the previous savings by compaction is lost and cannot affect whether or not DFSMShsm compacts the data set during subsequent migrations or backups.

Compaction considerations

Data sets sometimes exist on volumes in a format (compacted or uncompacted) that seems to conflict with the type of compaction specified with the SETSYS command. The following examples illustrate how this occurs.

DFSMShsm compacts data sets only when it copies them onto a DFSMShsm-owned volume from a level 0 volume.

If you specify . . .	Then . . .
SETSYS COMPACT (TAPEMIGRATION (ML2TAPE)) and SETSYS COMPACT (DASDMIGRATE TAPEMIGRATE)	DFSMShsm compacts data sets that migrate from level 0 volumes whether they migrate to DASD or whether they migrate directly to migration level 2 tape. DFSMShsm retains the compacted form when it moves data sets from migration level 1 DASD to migration level 2 tape.
SETSYS COMPACT (DASDMIGRATE NOTAPEMIGRATE)	DFSMShsm places both compacted and uncompacted data sets on migration level 2 tapes.
SETSYS COMPACT (DASDMIGRATE)	DFSMShsm compacts any data set migrating to migration level 1 DASD (or migration level 2 DASD, if DASD are used for ML2 volumes).
SETSYS COMPACT (NOTAPEMIGRATE)	DFSMShsm does not compact data sets that migrate from level 0 volumes directly to migration level 2 tapes. However, data sets migrating from level 1 volumes to level 2 tapes remain compacted; therefore, both compacted and uncompacted data sets can be on the tape.

Similarly, if you are not compacting data sets that migrate to DASD and you are compacting data sets that migrate directly to tape, both compacted and uncompacted data sets can migrate to level 2 tapes. The uncompacted data sets occur because the data sets are not compacted when they migrate to the migration level 1 DASD and the compaction is not changed when they later migrate to a migration level 2 tape. However, data sets migrating directly to tape are compacted.

If you specify . . .	Then . . .
SETSYS TAPEMIGRATION (DIRECT)	The DASDMIGRATE or NODASDMIGRATE subparameter of the SETSYS COMPACT command has no effect on DFSMShsm processing.

You can also have mixed compacted and uncompacted backup data sets and they, too, can be on either DASD or tape.

If you specify compaction for data sets backed up to DASD but no compaction for migrated data sets, any data set that migrates when it needs to be backed up is uncompacted when it is backed up from the migration volume.

Similarly, if you specify compaction for migrated data sets but no compaction for backed up data sets, a data set that migrates when it needs to be backed up migrates in compacted form. When the data set is backed up from the migration volume, it is backed up in its compacted form even though you specified no compaction for backup.

Data sets that are backed up to DASD volumes retain their compaction characteristic when they are spilled to tape. Thus, if you are not compacting data sets backed up to tape but you are compacting data sets backed up to DASD, you can have both compacted and uncompacted data sets on the same tapes. Data sets that are compacted and backed up to tape, likewise, can share tapes with uncompacted data sets that were backed up to DASD.

Optimum DASD blocking option

Each DASD device has an optimum block size for storing the maximum DFSMShsm data on each track. The default block size for DFSMShsm when it is storing data on its owned DASD devices is determined by the device type for each of the DFSMShsm-owned DASD devices to ensure that the maximum data is stored on each track of the device. For example, all models of 3390 DASD have the same track length, and therefore an optimum block size of 18KB (1KB equals 1024 bytes).

If you specify (not recommended) . . .	Then . . .
SETSYS NOOPTIMUMDASDBLOCKING	DFSMShsm uses a block size of 2KB for storing data on its owned DASD.

Data Set Reblocking

The purpose of reblocking is to make the most efficient use of available space.

If you specify . . .	Then . . .
SETSYS CONVERSION (REBLOCKTOANY)	DFSMShsm allows reblocking during recall or recovery to any device type supported by DFSMShsm, including target volumes of the same type as the source volume. This is the only parameter used by DFSMSdss.

Defining DFSMShsm reporting and monitoring

DFSMShsm produces information that can make the storage administrator, the operator, and the system programmer aware of what is occurring in the system. This information is provided in the form of activity logs, system console output, and entries in the System Management Facility (SMF) logs. You can specify a SETSYS command to control:

- The information that is stored in the activity logs
- The device type for the activity logs
- The messages that appear on the system console
- The type of output device for listings and reports
- Whether entries are made in the SMF logs

Figure 22 on page 91 is an example of the SETSYS commands that define a typical DFSMShsm reporting and monitoring environment.

```
/***********************************************************************/
/*    SAMPLE SETSYS COMMANDS THAT DEFINE A TYPICAL DFSMSHSM REPORTING  */
/*    AND MONITORING ENVIRONMENT                                       */
/***********************************************************************/
/*
SETSYS ACTLOGMSGLVL(EXCEPTIONONLY)
SETSYS ACTLOGTYPE(DASD)
SETSYS MONITOR (BACKUPCONTROLDATASET(80  ) -
                JOURNAL(80  ) -
                MIGRATIONCONTROLDATASET(80  ) -
                OFFLINECONTROLDATASET(80  ) -
                NOSPACE NOSTARTUP NOVOLUME)
SETSYS SYSOUT(A 1)
SETSYS SMF
/*
```

Figure 22. Sample Reporting and Monitoring Environment Definition for Typical DFSMShsm Environment

The activity logs are discussed in detail in Chapter 3, "DFSMShsm data sets," on page 9.

Controlling messages that appear on the system console

You can control the types of messages that appear at the system console by selecting the options for the SETSYS MONITOR command.

If you specify . . .	**Then . . .**
SETSYS MONITOR (MIGRATIONCONTROLDATASET(*threshold*) SETSYS MONITOR (BACKUPCONTROLDATASET(*threshold*)) SETSYS MONITOR (OFFLINECONTROLDATASET(*threshold*)) SETSYS MONITOR(JOURNAL(*threshold*))	DFSMShsm notifies the operator when a control data set is becoming full. You specify the threshold (percentage) of the allocated data set space that triggers a message.
SETSYS MONITOR(NOSPACE)	DFSMShsm does not issue volume space usage messages.
SETSYS MONITOR(NOSTARTUP)	DFSMShsm does not issue informational messages for startup progress.
SETSYS MONITOR(NOVOLUME)	DFSMShsm does not issue messages about data set activity on volumes it is processing.

For more information about the SETSYS command, see *z/OS DFSMShsm Storage Administration*.

Controlling the output device for listings and reports

The SYSOUT parameter controls where lists and reports are printed if the command that causes the list or report does not specify where it is to be printed. The default for this parameter is SYSOUT(A 1).

Controlling entries for the SMF logs

You determine if DFSMShsm writes System Management Facility (SMF) records to the SYS1.MANX and SYS1.MANY system data sets when you specify the SETSYS SMF or SETSYS NOSMF commands.

If you specify . . .	Then . . .
SETSYS SMF	DFSMShsm writes daily statistics records, function statistics records, and volume statistics records to the SYS1.MANX and SYS1.MANY system data sets.
SETSYS NOSMF	DFSMShsm does not write daily statistics records (DSRs), function statistics records (FSRs) or volume statistics records (VSRs) into the system data sets. For the formats of the FSR, DSR, and VSR records, see *z/OS DFSMShsm Diagnosis*.

Defining the tape environment

Chapter 10, "Implementing DFSMShsm tape environments," on page 189, contains information about setting up your tape environment, including discussions about SMS-managed tape libraries, tape management policies, device management policies, and performance management policies.

Defining the installation exits that DFSMShsm invokes

You determine the installation exits that DFSMShsm invokes when you specify the SETSYS(EXITON) or SETSYS(EXITOFF) commands. The installation exits can be dynamically loaded at startup by specifying them in your ARCCMD*xx* member in a PARMLIB.

Note: Examples of the DFSMShsm installation exits can be found in SYS1.SAMPLIB.

If you specify . . .	Then . . .
SETSYS EXITON(*exit*,*exit*,*exit***)**	The specified installation exits are immediately loaded and activated.
SETSYS EXITOFF(*exit*,*exit*,*exit***)**	The specified installation exits are immediately disabled and the storage is freed.
SETSYS EXITOFF(*exit1***)**	You modify and link-edit *exit1* and you then specify **SETSYS EXITON(***exit1***)**, DFSMShsm replaces the original exit1 with the newly modified exit1.

z/OS DFSMS Installation Exits describes the installation exits and what each exit accomplishes.

Controlling DFSMShsm control data set recoverability

The DFSMShsm journal data set records any activity that occurs to the DFSMShsm control data sets. By maintaining a journal, you ensure that damaged control data sets can be recovered by processing the journal against the latest backup copies of the control data sets.

If you specify . . .	Then . . .
SETSYS JOURNAL RECOVERY	DFSMShsm waits until the journal entry has been written into the journal before it updates the control data sets and continues processing.
SETSYS JOURNAL SPEED	DFSMShsm continues with its processing as soon as the journaling request has been added to the journaling queue. (Not recommended.)

For examples of data loss and recovery situations, refer to *z/OS DFSMShsm Storage Administration*.

Defining migration level 1 volumes to DFSMShsm

Whether you are implementing space management or availability management, you need migration level 1 volumes. Migration processing requires migration level 1 volumes as targets for data set migration. Backup processing requires migration level 1 volumes to store incremental backup and dump VTOC copy data sets. They may also be used as intermediate storage for data sets that are backed up by data set command backup.

Fast Replication backup requires migration level 1 volumes to store Catalog Information Data Sets. They may also be used as intermediate storage for data sets that are backed up by data set command backup.

Ensure that you include the ADDVOL command specifications for migration level 1 volumes in the ARCCMD*xx* member located in a PARMLIB, so that DFSMShsm recognizes the volumes at each startup. If ADDVOL commands for migration level 1 volumes are not in the ARCCMD*xx* member, DFSMShsm does not recognize that they are available unless you issue an ADDVOL command at the terminal for each migration level 1 volume. Figure 23 shows the sample ADDVOL commands for adding migration level 1 volumes to DFSMShsm control.

```
/***********************************************************************/
/*    SAMPLE ADDVOL COMMANDS FOR ADDING MIGRATION LEVEL 1 VOLUMES TO   */
/*    DFSMSHSM CONTROL                                                 */
/***********************************************************************/
/*
ADDVOL ML1001 UNIT(3390) -
       MIGRATION(MIGRATIONLEVEL1 -
       SMALLDATASETPACKING) THRESHOLD(90)

ADDVOL ML1002 UNIT(3390) -
       MIGRATION(MIGRATIONLEVEL1 -
       SMALLDATASETPACKING) THRESHOLD(90)

ADDVOL ML1003 UNIT(3390) -
       MIGRATION(MIGRATIONLEVEL1 -
       NOSMALLDATASETPACKING) THRESHOLD(90)
```

Figure 23. Example ADDVOL Commands for Adding Migration Level 1 Volumes to DFSMShsm Control

Parameters for the migration level 1 ADDVOL commands

The example below shows parameters used with the the MIGRATIONLEVEL1 parameter:

```
ADDVOL  1 ML1001  2 UNIT(3390) -
        3 MIGRATION(MIGRATIONLEVEL1 -
        SMALLDATASETPACKING) THRESHOLD(90)
```

- **1** - The first parameter of the ADDVOL command is a positional required parameter that specifies the volume serial number of the volume being added to DFSMShsm. In Figure 23, migration level 1 volumes are identified by volume serial numbers that start with ML1.
- **2** - The second parameter of the ADDVOL command is a required parameter that specifies the unit type of the volume. For our example, all ML1 volumes are 3390s.
- **3** - The third parameter is a required parameter that specifies that the volume is being added as a migration volume. This parameter has subparameters that

specify the kind of migration volume and the presence of a small-data-set-packing (SDSP) data set on the volume. If you specify SMALLDATASETPACKING, the volume must contain a VSAM key-sequenced data set to be used as the SDSP data set. See “DFSMShsm small-data-set-packing data set facility” on page 51 for details about how to allocate the SDSP data set.

The number of SDSP data sets defined must be at least equal to the maximum number of concurrent volume migration tasks that could be executing in your complex. Additional SDSPs are recommended for RECALL processing and ABARS processing and if some SDSPs should become full during migration.

- The THRESHOLD parameter in our ADDVOL command examples specifies the level of occupancy that signals the system to migrate data sets from migration level 1 volumes to migration level 2 volumes. If you want DFSMShsm to do automatic migration from level 1 to level 2 volumes, you must specify the occupancy thresholds for the migration level 1 volumes.

Note:

1. Automatic secondary space management determines whether to perform level 1 to level 2 migration by checking to see if any migration level 1 volume has an occupancy that is equal to or greater than its threshold. DFSMShsm migrates all eligible data sets from all migration level 1 volumes to migration level 2 volumes.
2. If the volume is being defined as a migration level 1 OVERFLOW volume then the threshold parameter is ignored. Use the SETSYS ML1OVERFLOW(THRESHOLD(nn)) command to specify the threshold for the entire OVERFLOW volume pool.
3. If you're adding volumes in an HSMplex environment, and those added volumes will be managed by each host in an HSMplex, then be sure to issue the ADDVOL on each system that will manage that volume.

For more information about level 1 to level 2 migration, see *z/OS DFSMShsm Storage Administration*.

In specifying the threshold parameter, you want to maintain equal free space on all of your migration level 1 volumes. If you use different device types for migration level 1 volumes, you must calculate the appropriate percentages that will make the same amount of free space available on each device type. For example, if you have a mixture of 3390 models 1 and 2, you might specify 88% for model 1 (92M) and 94% for model 2 (96M).

Using migration level 1 OVERFLOW volumes for migration and backup

An optional OVERFLOW parameter of the ADDVOL command lets you specify that OVERFLOW volumes are to be considered for backup or migration to migration level 1 when both the following are true:

- The file you are migrating or backing up is larger than a given size, as specified on the SETSYS ML1OVERFLOW(DATASETSIZE(*dssize*)) command
- DFSMShsm cannot allocate enough space on a NOOVERFLOW volume by selecting either the least used volume or the volume with the most free space.

Note that DFSMShsm will use OVERFLOW ML1 volumes for the following backup functions:

- Inline backup
- HBACKDS and BACKDS commands

- ARCHBACK macro for data sets larger than dssize K bytes

You can specify the OVERFLOW parameter as follows:

```
ADDVOL ML1003 UNIT(3390) -
      MIGRATION(MIGRATIONLEVEL1 OVERFLOW)
```

Related reading: For more information about the ADDVOL command and the SETSYS command, see *z/OS DFSMShsm Storage Administration*.

User or system data on migration level 1 volumes

Migration level 1 volumes, once defined to DFSMShsm, are known and used as *DFSMShsm-owned* volumes. That expression implies, among other things, that when DFSMShsm starts using such a volume, it determines the space available and creates its own record of that free space. For reasons of performance, DFSMShsm maintains that record as it creates and deletes its own migration copies, backup versions, and so on; DFSMShsm does not keep scanning the VTOC to see what *other* data sets may have been added or deleted.

Restrictions: The point is that your installation can store certain types of user or system data on migration level 1 volumes as long as you keep certain restrictions in mind:
- Such data sets *cannot* be SMS-managed, because these volumes cannot be SMS-managed.
- Once such a data set is allocated, do not change its size during an DFSMShsm startup.
- Do not request DFSMShsm to migrate or (except perhaps as part of a full dump of such a volume) back up such data sets.

Given that you maintain these restrictions, you can gain certain advantages by sharing these volumes with non-DFSMShsm data:
- A given amount of level-1 storage for DFSMShsm can be spread across more volumes, reducing volume contention.
- Since only one SDSP data set can be defined per volume, the number of such data sets can be increased.

Defining the common recall queue environment

DFSMShsm supports an HSMplex-wide common recall queue (CRQ). This CRQ balances recall workload across the HSMplex. This queue is implemented through the use of a coupling facility (CF) list structure. For an overview of the CRQ environment, refer to the *z/OS DFSMShsm Storage Administration*.

Updating the coupling facility resource manager policy for the common recall queue

The CRQ function requires that an HSMplex resides in a Parallel Sysplex® configuration. To fully utilize this function, allocate the list structure in a CF that supports the system-managed duplexing rebuild function. Before DFSMShsm can use the common recall queue, the active coupling facility resource management (CFRM) policy must be updated to include the CRQ definition. You can use the following information (see Table 7 on page 96) to define the CRQ and update the CFRM policy:

Table 7. Information that can be used to define the CRQ and update the CFRM policy.

Type	Information
Requirements	The structure name that must be defined in the active CFRM policy is 'SYSARC_'*basename*'_RCL', where *basename* is the base name specified in SETSYS COMMONQUEUE(RECALL(CONNECT(*basename*))). *basename* must be exactly five characters.
	The minimum CFLEVEL is eight. If the installation indicates that the structure must be duplexed, the system attempts to allocate the structure on a CF with a minimum of CFLEVEL=12.
	DFSMShsm does *not* specify size parameters when it connects to the CRQ. Size parameters must be specified in the CFRM policy. Refer to "Determining the structure size of the common recall queue" for a list of recommended structure sizes and the maximum number of concurrent recalls
	Because the list structure implements locks, the CF maintains an additional cross-system coupling facility (XCF) group in relation to this structure. Make sure that your XCF configuration can support the addition of another group.
	When implementing a CRQ environment, all hosts sharing a unique queue should be within the same SMSplex, have access to the same catalogs and DASD, and have common RACF configurations. The system administrator must ensure that all hosts connected to the CRQ are capable of recalling any migrated data set that originated from any other host that is connected to the same CRQ.
Recommendations	Nonvolatility is recommended, but not required. For error recovery purposes, each host maintains a local copy of each recall MWE that it places on the CRQ.
	CF failure independence is strongly recommended. For example, do not allocate the CRQ in a CF that is on an z/OS image that is on the same processor as another z/OS image with a system running a DFSMShsm that is using that CRQ.
Useful Information	Each CRQ is contained within a single list structure.
	A host can only connect to one CRQ at a time.
	DFSMShsm supports the alter function, including RATIO alters, and system-managed rebuilds. DFSMShsm does not support user-managed rebuilds. **Note:** System-managed rebuilds do not support the REBUILDPERCENT option.
	DFSMShsm supports the system-managed duplexing rebuild function.
	The CRQ is a persistent structure with nonpersistent connections. The structure remains allocated even if all connections have been deleted.

Determining the structure size of the common recall queue

The common recall queue needs to be sized such that it can contain the maximum number of concurrent recalls that may occur. Due to the dynamic nature of recall activity, there is no exact way to determine what the maximum number of concurrent recall requests may be.

Guideline: Use an INITSIZE value of 5120KB and a SIZE value of 10240KB.

A structure of this initial size is large enough to manage up to 3900 concurrent recall requests with growth up to 8400 concurrent recalls. These values should be large enough for most environments. Table 8 shows the maximum number of recalls that may be contained in structures of various sizes. No structure of less than 2560KB should be used.

Note: The maximum number of recall requests that may be contained within a structure is dependent on the number of requests that are from a unique Migration Level 2 tape. The figures shown in Table 8 are based on 33% of the recall requests requiring a unique ML2 tape. If fewer tapes are needed, then the structure will be able to contain more recall requests than is indicated.

Table 8. Maximum Concurrent Recalls

Structure Size	Maximum Concurrent Recalls
2560KB	1700
5120KB	3900
10240KB	8400
15360KB	12900

It should be recognized that the utilization percentage of the common recall queue will be low most of the time. This is because the average number of concurrent requests will be much lower than the maximum number of concurrent requests. In order to be prepared for a high volume of unexpected recall activity, the common recall queue structure size must be larger than the size needed to contain the average number of recall requests.

Altering the list structure size

DFSMShsm monitors how full a list structure has become. When the structure becomes 95% full, DFSMShsm no longer places recall requests onto the CRQ, but routes all new requests to the local queues. Routing recall requests to the CRQ resumes once the structure drops below 85% full. The structure is not allowed to become 100% full so that requests that are in-process can be moved between lists within the structure without failure. When the structure reaches maximum capacity, the storage administrator can increase the size by altering the structure to a larger size or by rebuilding it. A rebuild must be done if the maximum size has already been reached. (The maximum size limit specified in the CFRM policy must be increased before the structure is rebuilt). You can use the CF services structure full monitoring feature to monitor the structure utilization of the common recall queue.

How to alter the common recall queue list structure size

Initiate alter processing using the SETXCF START,ALTER command. Altering is a nondisruptive method for changing the size of the list structure. Alter processing can increase the size of the structure up to the maximum size specified in the CFRM policy. The SETXCF START,ALTER command can also decrease the size of a structure to the specified MINSIZE or default to the value specified in the CFRM policy.

How to rebuild the common recall queue list structure size

DFSMShsm supports the system-managed duplexing rebuild function. DFSMShsm does not support user-managed rebuilds.

Note: The coupling facility auto rebuild function does not support the use of REBUILDPERCENT. If the system rebuild function is not available because the structure was not allocated on a coupling facility that supports it, and a user needs to increase the maximum size of the structure or remedy a number of lost connections, then the user has to reallocate the structure.

Perform the following steps to reallocate the structure:

1. Disconnect all the hosts from the structure using the SETSYS COMMONQUEUE(RECALL(DISCONNECT)) command.
2. Deallocate the structure using the SETXCF FORCE command.
3. Reallocate the structure using the SETSYS COMMONQUEUE(RECALL(CONNECT(*basename*))) command.

Rule: If the intent of the rebuild is to increase the maximum structure size, you must update the CFRM policy before you perform the above steps.

Defining the common dump queue environment

DFSMShsm supports an HSMplex-wide common dump queue (CDQ). With CDQ, dump requests are distributed to a group of hosts for processing. This increases the number of available tasks to perform the work and improves throughput by distributing the workload rather than concentrating it on a single host's address space. For an overview of the CDQ environment and a description of how to define it, refer to common dump queue in *z/OS DFSMShsm Storage Administration*.

Defining common SETSYS commands

The following example shows typical SETSYS commands for an example system. Each of the parameters in "Defining common SETSYS commands" can be treated as a separate SETSYS command with the cumulative effect of a single SETSYS command. This set of SETSYS commands becomes part of the ARCCMD*xx* member pointed to by the DFSMShsm startup procedure.

```
/***********************************************************************/
/*      SAMPLE SETSYS COMMANDS THAT DEFINE THE MVS ENVIRONMENT       */
/***********************************************************************/
/*
SETSYS JES2
SETSYS CSALIMITS(MAXIMUM(100) ACTIVE(90) INACTIVE(30) MWE(4))
SETSYS NOREQUEST
SETSYS USERDATASETSERIALIZATION
SETSYS NOSWAP
SETSYS MAXABARSADDRESSSPACE(1)
/*
/***********************************************************************/
/*     SAMPLE SETSYS COMMANDS THAT DEFINE THE DFSMSHSM SECURITY       */
/***********************************************************************/
/*
SETSYS NOACCEPTPSCBUSERID
SETSYS NOERASEONSCRATCH
SETSYS NORACFIND
/*
/***********************************************************************/
/* SAMPLE SETSYS COMMANDS THAT DEFINE THE DATA FORMATS                 */
/***********************************************************************/
/*
SETSYS COMPACT(DASDMIGRATE NOTAPEMIGRATE DASDBACKUP NOTAPEBACKUP)
SETSYS COMPACTPERCENT(30)
SETSYS OBJECTNAMES(OBJECT,LINKLIB)
SETSYS SOURCENAMES(ASM,PROJECT)
```

```
SETSYS OPTIMUMDASDBLOCKING
SETSYS CONVERSION(REBLOCKTOANY)
SETSYS TAPEHARDWARECOMPACT
/*
/***********************************************************************/
/* SAMPLE SETSYS COMMANDS THAT DEFINE DFSMSHSM REPORTING AND           */
/* MONITORING ENVIRONMENT                                              */
/***********************************************************************/
/*
SETSYS ACTLOGMSGLVL(EXCEPTIONONLY)
SETSYS ACTLOGTYPE(DASD)
SETSYS MONITOR(BACKUPCONTROLDATASET(80  ) -
               JOURNAL(80  ) -
               MIGRATIONCONTROLDATASET(80  ) -
               OFFLINECONTROLDATASET(80  ) -
               NOSPACE NOSTARTUP NOVOLUME)
SETSYS SYSOUT(A 1)
SETSYS SMF
/*
/***********************************************************************/
/* SAMPLE SETSYS COMMANDS THAT DEFINE THE EXITS DFSMSHSM USES          */
/***********************************************************************/
/*
SETSYS EXITON(CD)
/*
/***********************************************************************/
/* SAMPLE SETSYS COMMANDS THAT DETERMINE DFSMSHSM RECOVERABILITY       */
/***********************************************************************/
/*
SETSYS JOURNAL(RECOVERY)
/*
/***********************************************************************/
/*    SAMPLE SETSYS COMMAND TO CONNECT TO A COMMON RECALL QUEUE LIST   */
/*    STRUCTURE. TEST1 IS THE BASE NAME OF THE CRQ LIST STRUCTURE      */
/***********************************************************************/
/*
SETSYS COMMONQUEUE(RECALL(CONNECT(TEST1)))
/***********************************************************************/
/* SAMPLE SETSYS COMMAND THAT SPECIFIES DATA SET SIZE AT WHICH AN      */
/* 0VERFLOW VOLUME IS PREFERRED FOR MIGRATION OR BACKUP                */
/***********************************************************************/
/*
SETSYS ML1OVERFLOW(DATASETSIZE(2000000))
/*
/***********************************************************************/
/* SAMPLE SETSYS COMMAND THAT SPECIFIES THE THRESHOLD OF ML1 OVERFLOW  */
/* VOLUME POOL SPACE FILLED BEFORE MIGRATION TO ML2 DURING SECONDARY   */
/* SPACE MANAGEMENT                                                    */
/***********************************************************************/
/*
SETSYS ML1OVERFLOW(THRESHOLD(80))
/*
```

Chapter 6. DFSMShsm starter set

For the customer who uses DFSMShsm in an SMS environment, the DFSMShsm starter set is shipped with the DFSMS licensed program. The DFSMShsm starter set complements the DFSMShsm distribution tape and contains sample implementation jobs and procedures.

If, as you edit or browse the starter set, you need more detailed information about the ADDVOL, DEFINE, ONLYIF, and SETSYS commands, see *z/OS DFSMShsm Storage Administration*. For information about defining your tape environment, see Chapter 10, "Implementing DFSMShsm tape environments," on page 189.

Basic starter set jobs

This section includes starter set objectives, recommended configurations, setup requirements, steps for running the starter set, and JCL listings.

Starter set objectives

This starter set has the following objectives:

- Reduce the workload to activate DFSMShsm in a DFSMS environment.
- Suggest parameters and procedures to adapt DFSMShsm to your environment.

Starter set configuration considerations

Review the following considerations when configuring and implementing the DFSMShsm starter set:

- Do not use the small-data-set-packing data set function, unless you have defined it on one or more level 1 migration volumes. See "Defining migration level 1 volumes to DFSMShsm" on page 93 for more detail.
- Do not define level 2 migration volumes. If you want to verify level 2 migration volumes, define them after you have the starter set running and after you have all other DFSMShsm functions running smoothly. "ARCCMD91" on page 127 is a sample job that initializes and identifies tapes (with the ADDVOL command) to DFSMShsm.
- Run the starter set on one *host* defined as *host* number one. Because this is the only *host* running DFSMShsm, *host* number one is also defined as the primary *host*. See the HOST= keyword of the DFSMShsm-started procedure in "Startup procedure keywords" on page 302 for more information.
- Back up the control data sets to tape.

Setup requirements

DFSMShsm requires the following information to setup and run the starter set in an SMS environment:

- The catalog, with its associated alias, which must be defined before attempting to run the installation verification procedure (IVP).
- An authorized user ID for the DFSMShsm-started procedure. (This user ID is also used as the high-level qualifier for the DFSMShsm-managed data sets. For the starter set, use the name DFHSM.)
- The user ID of a TSO user who is authorized to issue DFSMShsm AUTH commands.

- The type of JES on the system as either JES2 or JES3.
- The number of cylinders to allocate to any control data set.

  Guideline: Initially, allocate 10 cylinders to run the starter set. As you add more volumes and data sets to DFSMShsm control, you may need to calculate the size of the control data sets.
- The volume serial number and unit type of a volume for the log data sets.
- The name of the system master catalog.
- The job card parameters for each job.
- The volume serial number and unit type of a volume for the MCDS, BCDS, OCDS, and journal.
- The name of a user catalog for the data sets that DFSMShsm is to manage.
- The volume serial number and unit type of a volume for the user catalog.
- A storage class name for DFSMShsm control data sets, log, and journal data sets.
- A management class name for DFSMShsm data sets.
- A processing unit ID for the problem determination aid (PDA) facility.
- The volume serial number and unit type of a volume for the PDA data set.

Related reading

- Chapter 3, "DFSMShsm data sets," on page 9 (for calculating CDS sizes)
- "DFSMShsm log data set" on page 45
- "Migration control data set" on page 10
- "Backup control data set" on page 13
- "Offline control data set" on page 16
- "Journal data set" on page 19
- "DFSMShsm problem determination aid facility" on page 41

Steps for running the starter set

Before you begin: You need to know that the starter set is contained in member ARCSTRST in data set SYS1.SAMPLIB.

Perform the following steps to implement the DFSMShsm starter set. Figure 24 on page 103 contains boxed numbers that correspond to the numbered steps located below. As you read the following sequence of steps, refer to the figure for a graphic representation of the DFSMShsm starter set implementation.

1. Edit ARCSTRST and insert the correct job parameters in the job control statement.

   Result: See "HSMSTPDS" on page 106 for an example listing of this job.
2. Run ARCSTRST. ARCSTRST creates a data set called HSM.SAMPLE.CNTL.

   Result: See "HSM.SAMPLE.CNTL" on page 106 for a listing of the members in this data set.
3. Edit the member STARTER in the data set HSM.SAMPLE.CNTL, and globally change the parameters to reflect your computing system's environment.

   Result: See "STARTER" on page 107 for a list of parameters to change, and for an example listing of this job.
4. After the edit, run the member STARTER. STARTER creates the environment in which DFSMShsm runs. The environment includes ARCCMD00, a member of SYS1.PARMLIB, and DFSMSHSM, a member of SYS1.PROCLIB.

   Result: See "Starter set example" on page 109 for an example listing of the STARTER job.

5. If SMS is active, the DFSMShsm automatic class selection (ACS) routine should be in an active configuration. This prevents allocation of DFSMShsm-owned data sets on SMS-owned volumes.

   Result: For a sample ACS routine that directs DFSMShsm-owned data sets to non-SMS-managed storage, see Figure 16 on page 72.
6. Start DFSMShsm. Enter the following command: S DFSMSHSM. When you issue this command, the program calls ARCCTL.

You should now see messages on your screen that indicate that DFSMShsm has started. For an example of the messages that are displayed on the DFSMShsm startup screen (FVP), see Figure 25 on page 104.

You can stop DFSMShsm by entering the command: F DFSMSHSM,STOP. You will receive message ARC0002I, indicating that DFSMShsm has stopped successfully.

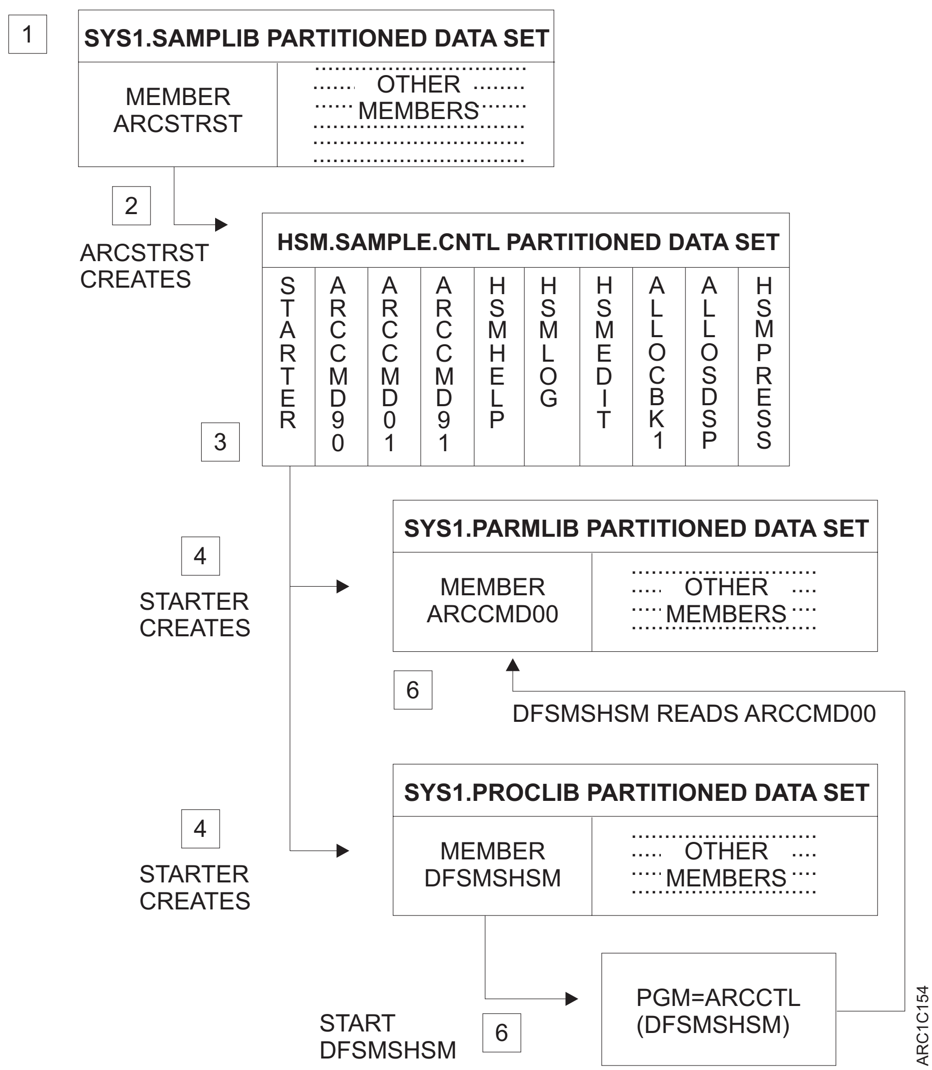


Figure 24. Starter Set Overview

```
$HASP100 DFSMSHSM ON STCINRDR
IEF695I START DFSMSHSM WITH JOBNAME DFSMSHSM IS ASSIGNED TO USER IBMUSER
 , GROUP SYS1
$HASP373 DFSMSHSM STARTED
ARC0041I MEMBER ARCSTR00 USED IN SYS1.PARMLIB
ARC0001I DFSMSHSM v.r.m STARTING HOST=1 IN 666
ARC0001I (CONT.) HOSTMODE=MAIN
ARC0037I DFSMSHSM PROBLEM DETERMINATION OUTPUT DATA 667
ARC0037I (CONT.) SETS SWITCHED, ARCPDOX=DFHSM.HSMPDOX,
ARC0037I (CONT.) ARCPDOY=DFHSM.HSMPDOY
ARC1700I DFSMSHSM COMMANDS ARE RACF PROTECTED
ARC0041I MEMBER ARCCMD00 USED IN SYS1.PARMLIB
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0270I PRIMARY SPACE MANAGEMENT CYCLE DEFINITION 701
ARC0270I (CONT.) SUCCESSFUL
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0270I SECONDARY SPACE MANAGEMENT CYCLE DEFINITION 706
ARC0270I (CONT.) SUCCESSFUL
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
```

Figure 25. Example of a z/OS Startup Screen (FVP) Part 1 of 2

```
 ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0270I BACKUP CYCLE DEFINITION SUCCESSFUL
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0270I DUMP CYCLE DEFINITION SUCCESSFUL
ARC0216I DUMPCLASS DEFINITION SUCCESSFUL, CLASS=SUNDAY 728
ARC0216I (CONT.) RC=0
ARC0216I DUMPCLASS DEFINITION SUCCESSFUL, 729
ARC0216I (CONT.) CLASS=QUARTERS RC=0
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0100I SETSYS COMMAND COMPLETED
ARC0038I RESOURCE MANAGER SUCCESSFULLY ADDED.  RETURN 749
ARC0038I (CONT.) CODE=00
ARC0008I DFSMSHSM INITIALIZATION SUCCESSFUL
```

Figure 26. Example of a z/OS Startup Screen (FVP) Part 2 of 2

Note: ARC0120I messages are issued only when ARCCMD00 has been modified to include ADDVOL commands. For additional information, see "Adapting and using the starter set" on page 124.

Modify the following starter set JCL listings to perform various jobs or functions.

JCL Listing
"HSMSTPDS" on page 106
"STARTER" on page 107
"ARCCMD90" on page 125
"ARCCMD01" on page 126
"ARCCMD91" on page 127
"HSMHELP" on page 128
"HSMLOG" on page 141
"HSMEDIT" on page 142
"ALLOCBK1" on page 142
"ALLOSDSP" on page 145

JCL Listing
"HSMPRESS" on page 147

HSMSTPDS

HSMSTPDS, an IEBUPDTE job found in member ARCSTRST of the SYS1.SAMPLIB. data set, creates the cataloged data set HSM.SAMPLE.CNTL. Refer to Figure 27 for a partial listing of the HSMSTPDS job.

```
//HSMSTPDS JOB ?JOBPARM
//HSMSTEP1 EXEC PGM=IEBGENER
//*
//*******************************************************************/
//* THESE SAMPLE DFSMSHSM PROGRAMS CREATE A PDS (HSM.SAMPLE.CNTL)    */
//* THAT CONTAINS THE FOLLOWING MEMBERS:                             */
//*                                                                  */
//* STARTER  - THE DFSMSHSM STARTER SET                              */
//* ARCCMD01 - SAMPLE ARCCMD MEMBER FOR ML2 TAPE PROCESSING          */
//* ARCCMD90 - SAMPLE ADDVOL COMMANDS FOR STARTER SET                */
//* ARCCMD91 - SAMPLE ADDVOL COMMANDS FOR ML2 TAPE PROCESSING        */
//* HSMHELP  - HELP TEXT FOR DFSMSHSM-AUTHORIZED COMMANDS            */
//* HSMLOG   - SAMPLE JOB TO PRINT THE LOG                           */
//* HSMEDIT  - SAMPLE JOB TO PRINT THE EDITLOG                       */
//* ALLOCBK1 - SAMPLE JOB TO ALLOCATE CDS BACKUP VERSION DATA SETS   */
//* ALLOSDSP - SAMPLE JOB TO ALLOCATE AN SDSP DATA SET               */
//* HSMPRESS - SAMPLE JOB TO REORGANIZE THE CONTROL DATA SETS        */
//*                                                            4@01D*/
//*                                                                  */
//* YOU CAN ESTABLISH AN OPERATING DFSMSHSM ON A SINGLE PROCESSOR    */
//* BY EDITING AND EXECUTING THE JOB CONTAINED IN THE MEMBER NAMED   */
//* STARTER. REFER TO THE DFSMSHSM IMPLEMENTATION AND CUSTOMIZATION  */
//* GUIDE - DFSMSHSM FOR INSTRUCTIONS ON USING THE DFSMSHSM STARTER  */
//* SET.                                                             */
//*******************************************************************/
//*
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD UNIT=SYSDA,
//            DSN=HSM.SAMPLE.CNTL(STARTER),
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120),
//            SPACE=(CYL,(1,1,10))
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM='$$'
//HSMALLOC JOB ?JOBPARM
           ...
           ...
   (STARTER data)
           ...
$$
//HSMSTEP2 EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD  SYSOUT=*
//SYSUT2   DD DSN=HSM.SAMPLE.CNTL,
//            DISP=OLD
//SYSIN    DD DATA,DLM='$$'
            ...
            ...
   (Data for the rest of the members in HSM.SAMPLE,CNTL)
$$
            ...
            ...
```

Figure 27. Partial Listing of Member HSMSTPDS

HSM.SAMPLE.CNTL

The following members of HSM.SAMPLE.CNTL help you adapt, configure, maintain, and monitor DFSMShsm. After you have run HSMSTPDS, found in member ARCSTRST, HSM.SAMPLE.CNTL contains the following members:.

Member	Description
STARTER	Contains the DFSMShsm starter set.
ARCCMD01	Contains DFSMShsm parameter specifications for using migration level 2 tape.
ARCCMD90	Contains sample ADDVOL commands for the starter set.
ARCCMD91	Contains sample ADDVOL commands for the starter set using migration level 2 tape.
HSMHELP	Contains a sample of the help file listing DFSMShsm commands and syntax.
HSMLOG	Contains a job to print the DFSMShsm log.
HSMEDIT	Contains a job to print the edit-log.
ALLOCBK1	Contains a sample job to allocate four control data set backup version data sets for each control data set.
ALLOSDSP	Contains a sample job to allocate a small data set packing data set.
HSMPRESS	Contains a job to reorganize the control data sets.

STARTER

STARTER is a job that you can edit and run to establish a basic DFSMShsm environment. Perform the following instructions:

1. Edit this member and globally change the following parameters.

   Restriction: Do not add any STEPCAT or JOBCAT statements to any DFSMShsm-started procedure (including this procedure). Unpredictable results can occur if these statements are specified.

   Parameter
   Description

   ?MCDSVOL
   Defines the volume serial number of the volume for the MCDS.

   ?MCDSUNT
   Defines the unit type of the MCDS volume.

   ?BCDSVOL
   Defines the volume serial number of the volume for the BCDS.

   ?BCDSUNT
   Defines the unit type for the BCDS volume.

   ?OCDSVOL
   Defines the volume serial number of the volume for the OCDS.

   ?OCDSUNT
   Defines the unit type for the OCDS volume.

   ?CDSSIZE
   Defines the number of cylinders allocated to any control data set (allocate 10 cylinders, initially, for the starter set).

   ?JRNLVOL
   Defines the volume serial number of the volume for the journal data set.

   ?JRNLUNT
   Defines the unit type for the journal volume.

?LOGVOL
Defines the volume serial number of the volume for the log data sets.

?LOGUNIT
Defines the unit type for the log volume.

?TRACEVOL
Defines the volume serial number of the problem determination aid trace data set.

?TRACEUNIT
Defines the unit type for the problem determination aid log volume.

?UCATNAM
Defines the name of the user catalog for the DFSMShsm data sets.

?UCATVOL
Defines the volume serial number of the volume for the user catalog.

?UCATUNT
Defines the unit type for the user catalog volume.

?CTLAUTH
Defines the user ID of a DFSMShsm-authorized user who can issue DFSMShsm AUTH commands.

?UID
Defines the authorized user ID for the DFSMShsm-started procedure. An authorized user ID must be from 1 to 7 characters in length. This ID is also the high-level qualifier for Small Data Set Packing data sets.

Note: UID authorization is valid only in a non-FACILITY class environment. Otherwise, DFSMShsm uses RACF FACILITY class profiles for authorization checking.

As a matter of convenience and simplicity for these examples, this ID is also used throughout this publication as the high-level qualifier for control data sets, logs, and PDA trace data sets. However, these data sets can have different high-level qualifiers, as needed to fit your naming standards.

?JESVER
Defines the job entry subsystem (JES) as either JES2 or JES3.

?JOBPARM
Defines the job card parameters.

?MCATNAM
Defines the name and password of the system master catalog.

?SCLOGNM
Defines the storage class name for the DFSMShsm log and journal data sets.

?SCCDSNM
Defines the storage class name for the DFSMShsm control data sets.

?MCDFHSM
Defines the management class name for DFSMShsm data sets.

?HOSTID
Defines the DFSMShsm host ID for the problem determination aid facility and for identifying the host to DFSMShsm.

?PRIMARY
: Defines whether or not the DFSMShsm host performs as a primary host. Contains a value of YES or NO.

?NEW
: Extension of CDS name for IMPORT (HSMPRESS).

2. During the edit, globally search for the character strings, "REMOVE THE NEXT ... " and determine if the following JCL statements apply to your environment. If these JCL statements do not apply, ensure that they are removed from the data set.
3. Also during the edit, search for the character string "DFSMSHSM AUTOMATIC FUNCTIONS" and determine if the material described in the comments applies to your environment.
4. If you want to add (ADDVOL command) non-SMS volumes (primary, migration, backup, and dump volumes) to your DFSMShsm environment, edit member ARCCMD90 to identify those volumes to DFSMShsm and append ARCCMD90 to ARCCMD00.

   If you want to migrate data sets to ML2 tape, edit members ARCCMD01 and ARCCMD91 and append them to ARCCMD00.
5. After editing STARTER, you can start the job.

Starter set example

The DFSMShsm startup procedure identifies the ARCCMD*xx* PARMLIB member containing the commands that define the DFSMShsm environment. It also identifies the startup procedure keywords. For a detailed discussion of the DFSMShsm ARCCMD*xx* PARMLIB member and the DFSMShsm startup procedure keywords, see Chapter 14, "DFSMShsm libraries and procedures," on page 297.

"Example of Starter Set " is an example listing of the STARTER member. As you review the starter set, you will notice that many of the tasks required to implement DFSMShsm are included in the following code samples.

Example of Starter Set

```
//*******************************************************************/
//* DFSMSHSM STARTER SET                                            */
//*                                                                 */
//* THIS JCL STREAM ESTABLISHES AN OPERATING DFSMSHSM ENVIRONMENT   */
//* FOR A NEW USER OF DFSMSHSM OPERATING IN AN SMS ENVIRONMENT, OR  */
//* FOR A USER WHO WANTS TO RUN THE FUNCTIONAL VERIFICATION         */
//* PROCEDURE (FVP) IN AN ENVIRONMENT THAT IS SEPARATE FROM THE     */
//* PRODUCTION ENVIRONMENT.  THE FVP IS FOUND IN ARCFVPST.          */
//*                                                                 */
//* EDIT THIS JCL FOR YOUR PROCESSING ENVIRONMENT.                  */
//*                                                                 */
//* YOU CAN DECREASE IMPLEMENTATION TIME BY MAKING GLOBAL CHANGES   */
//* TO THE FOLLOWING PARAMETERS.  YOU MAY HAVE TO MAKE OTHER CHANGES */
//* AS IDENTIFIED IN THE COMMENTS EMBEDDED IN THE JCL.              */
//*                                                                 */
//* IF YOU ALLOCATED SMS-MANAGED DATA SETS ON SPECIFIC VOLUMES,     */
//* ENSURE THAT YOU ASSOCIATE THOSE DATA SETS WITH THE GUARANTEED   */
//* SPACE ATTRIBUTE IN THEIR STORAGE CLASS DEFINITION.              */
//*                                                                 */
//* WE RECOMMEND THAT YOU DEFINE ALL DFSMSHSM DATA SETS WITH THE    */
//* GUARANTEED-SPACE ATTRIBUTE IN THEIR STORAGE CLASS DEFINITIONS.  */
//*                                                                 */
//* WE RECOMMENDED THAT YOU DEFINE ALL DFSMSHSM DATA SETS WITH THE  */
//* NO-MIGRATE AND NO-BACKUP ATTRIBUTES IN THEIR MANAGEMENT CLASS   */
//* DEFINITIONS.  YOU CAN PREVENT DFSMSHSM DATA SETS FROM MIGRATING */
//* OR BEING BACKED UP BY ASSIGNING THEM TO THE DBSTNDRD MANAGEMENT */
```

```
//* CLASS.                                                            */
//*                                                                   */
//* WE RECOMMEND THAT YOU DEFINE THE LOG AND JOURNAL DATA SETS WITH  */
//* A STORAGE CLASS DEFINITION FOR LOGGING OR FOR AUDIT TRAIL DATA   */
//* SETS BY DEFINING THEM WITH THE STORAGE CLASS DBLOG.               */
//*                                                                   */
//* WE RECOMMEND THAT YOU DEFINE THE DFSMSHSM CONTROL DATA SETS WITH */
//* A STORAGE CLASS DEFINITION THAT PROVIDES FAST RESPONSE BY         */
//* DEFINING THEM WITH THE STORAGE CLASS DBENHANC.                    */
//*                                                                   */
//* THE SMS CONSTRUCTS (STORAGE CLASSES, STORAGE GROUPS, MANAGEMENT  */
//* CLASSES, AND DATA CLASSES) ARE DISCUSSED IN THE STORAGE           */
//* ADMINISTRATION GUIDE FOR DFSMSDFP.                                */
//*******************************************************************/
//*                                                                   */
//* CHANGE THE FOLLOWING PARAMETERS FOR YOUR PROCESSING ENVIRONMENT. */
//*                                                                   */
//*******************************************************************/
//*  PARAMETER     PARAMETER DEFINITION
//*
//*  ?MCDSVOL   -  VOLUME SERIAL NUMBER OF THE MCDS VOLUME
//*  ?MCDSUNT   -  UNIT TYPE FOR MCDS VOLUME
//*  ?BCDSVOL   -  VOLUME SERIAL NUMBER OF THE BCDS VOLUME
//*  ?BCDSUNT   -  UNIT TYPE FOR BCDS VOLUME
//*  ?OCDSVOL   -  VOLUME SERIAL NUMBER OF THE OCDS VOLUME
//*  ?OCDSUNT   -  UNIT TYPE FOR OCDS VOLUME
//*  ?CDSSIZE   -  NUMBER OF CYLINDERS TO INITIALLY ALLOCATE FOR ANY
//*                CONTROL DATA SET
//*  ?JRNLVOL   -  VOLUME SERIAL NUMBER OF THE JOURNAL
//*  ?JRNLUNT   -  UNIT TYPE FOR JOURNAL VOLUME
//*  ?LOGVOL    -  VOLUME SERIAL NUMBER OF THE LOG VOLUME
//*  ?LOGUNIT   -  UNIT TYPE FOR LOG VOLUME
//*  ?TRACEVOL  -  VOLUME SERIAL NUMBER OF THE PROBLEM DETERMINATION
//*                AID VOLUME
//*  ?TRACEUNIT -  UNIT TYPE FOR THE PROBLEM DETERMINATION AID VOLUME
//*  ?UCATNAM   -  NAME OF THE USER CATALOG FOR THE DFSMSHSM DATA SETS.
//*  ?UCATVOL   -  VOLUME SERIAL NUMBER OF THE USER CATALOG VOLUME
//*  ?UCATUNT   -  UNIT TYPE FOR USER CATALOG VOLUME
//*  ?CTLAUTH   -  THE USER ID THAT CAN ISSUE DFSMSHSM AUTH COMMANDS
//*                (YOUR CONTROL-AUTHORIZED USER ID)
//*  ?UID       -  AUTHORIZED USER ID (1 - 7 CHARACTERS) FOR THE
//*                DFSMSHSM-STARTED PROCEDURE IN A NON-FACILITY CLASS
//*                ENVIRONMENT (SEE NOTE BELOW).  THIS IS THE
//*                HIGH-LEVEL QUALIFIER FOR DFSMSHSM DATA SETS.
//*  ?JESVER    -  THE JOB ENTRY SUBSYSTEM (JES); EITHER JES2 OR JES3
//*  ?JOBPARM   -  JOB CARD PARAMETERS
//*  ?SCLOGNM   -  STORAGE CLASS FOR DFSMSHSM LOG AND JOURNAL
//*  ?SCCDSNM   -  STORAGE CLASS NAME FOR DFSMSHSM CONTROL DATA SETS
//*  ?MCDFHSM   -  MANAGEMENT CLASS NAME FOR DFSMSHSM DATA SETS
//*  ?HOSTID    -  PROCESSING UNIT ID FOR THE PROBLEM DETERMINATION
//*                AID FACILITY AND FOR IDENTIFYING THE HOST TO
//*                DFSMSHSM
//*  ?PRIMARY   -  YES OR NO; DEFINES WHETHER OR NOT THE DFSMSHSM
//*                HOST PERFORMS AS A PRIMARY HOST
//*  ?NEW       -  EXTENSION OF CDS NAME FOR IMPORT (HSMPRESS)
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES WILL BE
//* USED FOR AUTHORIZATION CHECKING.)
//******************************************************************/
//*
//IDCAMS EXEC PGM=IDCAMS
//*
//******************************************************************/
//*  ENSURE THAT DFSMSHSM CONTROL DATA SETS, THE JOURNAL, AND ANY   */
//*  CONTROL DATA SET BACKUP COPIES ARE ON DIFFERENT VOLUMES FROM   */
//*  EACH OTHER.                                                     */
//*                                                                  */
```

```
//*  GIVE USERS WRITE ACCESS TO VSAM DATA SETS BY DEFINING VSAM     */
//*  DATA SETS WITH A SHAREOPTION OF (3 3).  IT IS THE USER'S       */
//*  RESPONSIBILITY TO PROTECT THE CONTROL DATA SETS AGAINST        */
//*  UNAUTHORIZED ACCESS.                                           */
//*                                                                 */
//*******************************************************************/
//*
//HSMMCDS  DD UNIT=?MCDSUNT,VOL=SER=?MCDSVOL,DISP=SHR
//HSMCAT   DD UNIT=?UCATUNT,DISP=SHR,VOL=SER=?UCATVOL
//*
//*******************************************************************/
//* REMOVE THE NEXT DD STATEMENT IF YOU DO NOT INTEND TO USE BACKUP */
//* AND DUMP.                                                       */
//*******************************************************************/
//*
//HSMBCDS  DD UNIT=?BCDSUNT,VOL=SER=?BCDSVOL,DISP=SHR
//*
//*******************************************************************/
//* REMOVE THE NEXT DD STATEMENT IF YOU DO NOT INTEND TO USE TAPE   */
//* VOLUMES FOR DAILY BACKUP VOLUMES, SPILL BACKUP VOLUMES, OR      */
//* MIGRATION LEVEL 2 VOLUMES.                                      */
//*******************************************************************/
//*
//HSMOCDS  DD UNIT=?OCDSUNT,VOL=SER=?OCDSVOL,DISP=SHR
//*
//SYSIN    DD *
 /*                                                                 */
 /*******************************************************************/
 /* THIS JOB ALLOCATES AN INTEGRATED CATALOG FACILITY (ICF) CATALOG */
 /* AND ITS ASSOCIATED ALIAS "?UID".                                */
 /*                                                                 */
 /*  ****** INTEGRATED CATALOG FACILITY CATALOG REQUIRED *******    */
 /*                                                                 */
 /* THIS JOB ALLOCATES A USER CATALOG FOR THE DFSMSHSM CONTROL DATA */
 /* SETS (CDS). SEE THE SECTION "DFSMSHSM DATA SETS" IN THE         */
 /* DFSMSHSM IMPLEMENTATION AND CUSTOMIZATION GUIDE.                */
 /*******************************************************************/
 /*                                                                 */
  DEFINE UCAT(NAME(?UCATNAM) -
         CYLINDERS(1 1) VOLUME(?UCATVOL) -
         FILE(HSMCAT) FREESPACE(10 10) -
         RECORDSIZE(4086 4086) -
         ICFCATALOG)
  IF MAXCC = 0 THEN DO
    DEFINE  ALIAS(NAME(?UID) RELATE(?UCATNAM))
  END
 /*                                                              */
 /****************************************************************/
 /*  THIS PROCEDURE ASSUMES A SINGLE CLUSTER MCDS.  IF MORE THAN */
 /*  ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER  */
 /*  CDS.                                                        */
 /****************************************************************/
 /*                                                              */
 IF MAXCC = 0 THEN DO
   DEFINE CLUSTER (NAME(?UID.MCDS) VOLUMES(?MCDSVOL) -
        CYLINDERS(?CDSSIZE) FILE(HSMMCDS) -
        STORCLAS(?SCCDSNM) -
        MGMTCLAS(?MCDFHSM) -
        RECORDSIZE(435 2040) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(?UID.MCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(?UID.MCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
 END
```

```
 /*                                                              */
 /****************************************************************/
 /* REMOVE THE NEXT DEFINE COMMAND IF YOU DO NOT                 */
 /* INTEND TO USE BACKUP, DUMP OR AGGREGATE BACKUP AND RECOVERY. */
 /*                                                              */
 /* THIS PROCEDURE ASSUMES A SINGLE CLUSTER BCDS.  IF MORE THAN  */
 /* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER   */
 /* CDS.                                                         */
 /*                                                              */
 /* IT'S RECOMMENDED THAT YOU SHOULD SPECIFY RECORDSIZE(334 2093)*/
 /* AND CISIZE(12288) WHEN CREATING UP TO 29 BACKUP VERSIONS     */
 /* OR RECORDSIZE(334 6544) AND CISIZE(12288) IF UP TO           */
 /* 100 BACKUP VERSIONS WILL BE KEPT OR IF FAST REPLICATION IS   */
 /* BEING USED (FRBACKUP).                                       */
 /*                                                              */
 /****************************************************************/
 /*                                                              */
 IF MAXCC = 0 THEN DO
   DEFINE CLUSTER (NAME(?UID.BCDS) VOLUMES(?BCDSVOL) -
        CYLINDERS(?CDSSIZE) FILE(HSMBCDS) -
        STORCLAS(?SCCDSNM) -
        MGMTCLAS(?MCDFHSM) -
        RECORDSIZE(334 6544) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(?UID.BCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(?UID.BCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
 END
 /*
 /****************************************************************/
 /* REMOVE THE NEXT DEFINE COMMAND IF YOU DO NOT                 */
 /* INTEND TO USE TAPES FOR DAILY BACKUP, SPILL BACKUP, OR       */
 /* MIGRATION LEVEL 2 PROCESSING.                                */
 /*                                                              */
 /*                                                              */
 /* IT IS RECOMMENDED THAT YOU SPECIFY                           */
 /*   RECORDSIZE(1800 2040) WHEN NOT USING EXTENDED TTOCS AND    */
 /*   RECORDSIZE(1080 6144) WHEN USING EXTENDED TTOCS.           */
 /*                                                              */
 /* NOTE: YOU CAN ONLY USE EXTENDED TTOCS IF ALL OF YOUR         */
 /* DFSMSHSM HOSTS ARE AT Z/OS DFSMSHSM V1R7 OR LATER.           */
 /*                                                              */
 /* THE OCDS MAY NOT EXCEED 1 VOLUME.                            */
 /****************************************************************/
 /*
 IF MAXCC = 0 THEN DO
   DEFINE CLUSTER (NAME(?UID.OCDS) VOLUMES(?OCDSVOL) -
        CYLINDERS(?CDSSIZE) FILE(HSMOCDS) -
        STORCLAS(?SCCDSNM) -
        MGMTCLAS(?MCDFHSM) -
        RECORDSIZE(1800 2040) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(?UID.OCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(?UID.OCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
 END
//SYSPRINT DD SYSOUT=*
//*
//***************************************************************/
//*  ALLOCATE AND CATALOG THE DFSMSHSM LOG, EDIT LOG, AND JOURNAL*/
//*  ON AN "SMS" VOLUME.                                         */
```

```
//*****************************************************************/
//*
//LOGALC   EXEC PGM=IEFBR14
//HSMLOGX  DD DSN=?UID.HSMLOGX1,DISP=(,CATLG),UNIT=?LOGUNIT,
//   VOL=SER=?LOGVOL,SPACE=(CYL,(3)),STORCLAS=?SCLOGNM,
//   MGMTCLAS=?MCDFHSM
//HSMLOGY  DD DSN=?UID.HSMLOGY1,DISP=(,CATLG),UNIT=?LOGUNIT,
//   VOL=SER=?LOGVOL,SPACE=(CYL,(3)),STORCLAS=?SCLOGNM,
//   MGMTCLAS=?MCDFHSM
//EDITLOG  DD DSN=?UID.EDITLOG,DISP=(,CATLG),UNIT=?LOGUNIT,
//   VOL=SER=?LOGVOL,SPACE=(CYL,(2)),STORCLAS=?SCLOGNM,
//   MGMTCLAS=?MCDFHSM
//*
//****************************************************************/
//*  THE JOURNAL MUST NOT EXCEED 1 VOLUME, MAY NOT HAVE          */
//*  SECONDARY ALLOCATION, AND MUST BE ALLOCATED CONTIGUOUS.     */
//****************************************************************/
//*
//JOURNAL DD DSN=?UID.JRNL,DISP=(,CATLG),UNIT=?JRNLUNT,
//   VOL=SER=?JRNLVOL,SPACE=(CYL,(5),,CONTIG),STORCLAS=?SCLOGNM,
//   MGMTCLAS=?MCDFHSM
//*
//****************************************************************/
//*  ALLOCATE THE PROBLEM DETERMINATION AID (PDA) LOG ON "SMS"   */
//*  OR ON 'NONSMS' VOLUME.  USE THE JCL BELOW FOR NONSMS        */
//*  OR ADJUST THE BELOW TO MATCH THE JCL ABOVE FOR THE LOG      */
//*  BY ADDING STORCLAS AND MGMTCLASS.                           */
//*  REMOVE THE NEXT TWO DD CARDS IF YOU DO NOT PLAN TO USE PDA. */
//****************************************************************/
//*
//ARCPDOX  DD DSN=?UID.HSMPDOX,DISP=(,CATLG),VOL=SER=?TRACEVOL,
//         UNIT=?TRACEUNIT,SPACE=(CYL,(20,2))
//ARCPDOY  DD DSN=?UID.HSMPDOY,DISP=(,CATLG),VOL=SER=?TRACEVOL,
//         UNIT=?TRACEUNIT,SPACE=(CYL,(20,2))
//HSMPROC  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=SYS1.PROCLIB,DISP=SHR
//SYSIN    DD DATA,DLM='$A'
./ ADD NAME=DFSMSHSM
//*
//****************************************************************/
//*  THE JOURNAL MUST NOT EXCEED 1 VOLUME, MAY NOT HAVE          */
//*  SECONDARY ALLOCATION, AND MUST BE ALLOCATED CONTIGUOUS.     */
//****************************************************************/
//*
//JOURNAL DD DSN=?UID.JRNL,DISP=(,CATLG),UNIT=?JRNLUNT,
//   VOL=SER=?JRNLVOL,SPACE=(CYL,(5),,CONTIG),STORCLAS=?SCLOGNM,
//   MGMTCLAS=?MCDFHSM
//*
//****************************************************************/
//*  ALLOCATE THE PROBLEM DETERMINATION AID (PDA) LOG ON "SMS"   */
//*  OR ON 'NONSMS' VOLUME.  USE THE JCL BELOW FOR NONSMS        */
//*  OR ADJUST THE BELOW TO MATCH THE JCL ABOVE FOR THE LOG      */
//*  BY ADDING STORCLAS AND MGMTCLASS.                           */
//*  REMOVE THE NEXT TWO DD CARDS IF YOU DO NOT PLAN TO USE PDA. */
//****************************************************************/
//*
//ARCPDOX  DD DSN=?UID.HSMPDOX,DISP=(,CATLG),VOL=SER=?TRACEVOL,
//         UNIT=?TRACEUNIT,SPACE=(CYL,(20,2))
//ARCPDOY  DD DSN=?UID.HSMPDOY,DISP=(,CATLG),VOL=SER=?TRACEVOL,
//         UNIT=?TRACEUNIT,SPACE=(CYL,(20,2))
//HSMPROC  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=SYS1.PROCLIB,DISP=SHR
//SYSIN    DD DATA,DLM='$A'
./ ADD NAME=DFSMSHSM
//*
```

```
//*******************************************************************/
//*                  DFSMSHSM START PROCEDURE                        */
//*                                                                  */
//* YOU CAN DUPLICATE AND RENAME THE FOLLOWING PROCEDURE FOR OTHER  */
//* PROCESSORS IN A MULTIPLE-PROCESSING-UNIT ENVIRONMENT.           */
//* ENSURE THAT YOU CHANGE THE CMD= AND HOST= KEYWORDS              */
//* ENSURE THAT YOU CHANGE THE HIGH-LEVEL QUALIFIER FOR THE         */
//* ARCLOGX AND ARCLOGY DATA SET NAMES.                             */
//* KEYWORD DEFINITIONS:                                             */
//*    CMD=00                SPECIFY WHICH PARMLIB COMMAND MEMBER  */
//*    STR=xx                REPLACE xx WITH LAST TWO CHARACTERS   */
//*                          OF THE SYS1.PARMLIB(ARCSTRxx)         */
//*                          MEMBER THAT YOU CREATE PRIOR TO       */
//*                          STARTING DFSMSHSM                     */
//*    EMERG=YES|NO          START HSM IN EMERGENCY MODE           */
//*    SIZE=0M               REGION SIZE FOR DFSMSHSM              */
//*    LOGSW=YES|NO          SWITCH LOGS AT STARTUP                */
//*    STARTUP=YES|NO        STARTUP INFO PRINT AT STARTUP         */
//*    PDA=YES|NO            BEGIN PDA TRACING AT STARTUP          */
//*    HOST=X                SPECIFY HOSTID                        */
//*    PRIMARY=YES|NO        SPECIFY PRIMARY HOST                  */
//*    HOSTMODE=MAIN|AUX     INDICATE IF THIS IS A MAIN OR AUX HOST*/
//*    DDD=50                MAX DYNAMICALLY ALLOCATED DATASETS    */
//*    RNAMEDSN=YES|NO       USE EXTENDED RESOURCE NAMES           */
//*    CDSQ=YES|NO           SERIALIZE CDSs WITH GLOBAL ENQUEUES   */
//*    CDSR=YES|NO           SERIALIZE CDSs WITH VOLUME RESERVES   */
//*    CDSSHR=YES|NO|RLS     SPECIFY CDS SHARABILITY               */
//*    RESTART=(A,B)         RESTART DFSMSHSM AFTER ABEND          */
//*    CELLS=(200,100,100,50,20)  SIZES OF CELLPOOLS               */
//*    UID=HSM               DFSMSHSM-AUTHORIZED USER ID.  ALSO    */
//*                          USED FOR HLQ OF HSM DATASETS BUT NOT  */
//*                          REQUIRED.                             */
//*******************************************************************/
//* IF ALL OF THE DFSMSHSM STARTUP PROCEDURE KEYWORDS ARE NEEDED,   */
//* TOTAL LENGTH WILL EXCEED THE 100-BYTE LIMIT, IN WHICH CASE      */
//* YOU SHOULD USE THE KEYWORD STR=XX IN PARM= TO IDENTIFY THE      */
//* PARMLIB MEMBER CONTAINING THE ADDITIONAL KEYWORDS AND PARMS.    */
//*******************************************************************/
//DFSMSHSM   PROC CMD=00,     USE PARMLIB MEMBER ARCCMD00 FOR CMDS
//           STR=00,          PARMLIB MEMBER FOR STARTUP PARMS
//           EMERG=NO,        SETS HSM INTO NON-EMERGENCY MODE
//           CDSQ=YES,        CDSs SERIALIZED WITH ENQUEUES
//           PDA=YES,         PROBLEM DETERMINATION AID
//           SIZE=0M,         REGION SIZE FOR DFSMSHSM
//           DDD=50,          MAX DYNAMICALLY ALLOCATED DATASETS
//           HOST=?HOSTID,    PROC.UNIT ID AND LEVEL FUNCTIONS
//           PRIMARY=?PRIMARY LEVEL FUNCTIONS
//*******************************************************************/
//DFSMSHSM  EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//          PARM=('EMERG=&EMERG','CMD=&CMD','CDSQ=&CDSQ',
//          'UID=?UID','PDA=&PDA','HOST=&HOST','STR=&STR',
//          'PRIMARY=&PRIMARY')
//*****************************************************************/
//*   HSMPARM DD must be deleted from the JCL or made into a      */
//*   comment to use Concatenated Parmlib Support                 */
//*****************************************************************/
//HSMPARM  DD DSN=SYS1.PARMLIB,DISP=SHR
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//SYSPRINT DD SYSOUT=A,FREE=CLOSE
//SYSUDUMP DD SYSOUT=A
//*
//*****************************************************************/
//* THIS PROCEDURE ASSUMES A SINGLE CLUSTER MCDS.  IF MORE THAN   */
//* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER    */
//* CDS.                                                          */
//*****************************************************************/
```

```
//*
//MIGCAT   DD DSN=?UID.MCDS,DISP=SHR
//JOURNAL  DD DSN=?UID.JRNL,DISP=SHR
//ARCLOGX  DD DSN=?UID.HSMLOGX1,DISP=OLD
//ARCLOGY  DD DSN=?UID.HSMLOGY1,DISP=OLD
//ARCPDOX  DD DSN=?UID.HSMPDOX,DISP=OLD
//ARCPDOY  DD DSN=?UID.HSMPDOY,DISP=OLD
//*
//*****************************************************************/
//* REMOVE THE NEXT DD STATEMENT IF YOU DO NOT INTEND TO USE      */
//* BACKUP AND DUMP.                                              */
//*                                                               */
//* THIS PROCEDURE ASSUMES A SINGLE CLUSTER BCDS.  IF MORE THAN   */
//* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER    */
//* CDS.                                                          */
//*****************************************************************/
//*
//BAKCAT   DD DSN=?UID.BCDS,DISP=SHR
//*
//*****************************************************************/
//* REMOVE THE NEXT DD STATEMENT IF YOU DO NOT INTEND TO USE TAPES*/
//* FOR DAILY BACKUP, SPILL BACKUP OR MIGRATION LEVEL 2           */
//* PROCESSING.                                                   */
//*                                                               */
//* THE OCDS MAY NOT EXCEED 1 VOLUME.                             */
//*****************************************************************/
//*
//OFFCAT   DD DSN=?UID.OCDS,DISP=SHR
./ ADD NAME=DFHSMABR
//*
//*****************************************************************/
//*     ABARS SECONDARY ADDRESS SPACE STARTUP PROCEDURE           */
//*****************************************************************/
//*
//DFHSMABR PROC
//DFHSMABR EXEC PGM=ARCWCTL,REGION=0M
//SYSUDUMP DD SYSOUT=A
//MSYSIN DD DUMMY
//MSYSOUT DD SYSOUT=A
$A
//HSMPROC  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSIN    DD DATA,DLM='$A'
./ ADD NAME=ARCCMD00
 /****************************************************************/
 /*             DFSMSHSM STARTUP COMMAND MEMBER                  */
 /*             WITH ONLY LEVEL 1 MIGRATION                      */
 /****************************************************************/
 /****************************************************************/
 /*            DFSMSHSM AUTOMATIC FUNCTIONS                      */
 /*                                                              */
 /* *********** AUTOBACKUPSTART,  AUTODUMPSTART, *****************/
 /* ********** AUTOMATIC PRIMARY SPACE MANAGEMENT, ***************/
 /* ********* AUTOMATIC SECONDARY SPACE MANAGEMENT. **************/
 /*                                                              */
 /* THE AUTOMATIC DFSMSHSM FUNCTIONS SPECIFIED IN THE FOLLOWING  */
 /* SETSYS COMMANDS CONTAIN ZEROS FOR START, LATE START, AND      */
 /* TIMES. THUS, NO AUTOMATIC FUNCTIONS ARE ACTIVATED WHEN       */
 /* DFSMSHSM IS STARTED ON YOUR SYSTEM. TO ACTIVATE AUTOMATIC    */
 /* FUNCTIONS, CHANGES THE TIMES TO VALUES THAT ARE APPROPRIATE  */
 /* FOR YOUR SYSTEM.                                             */
 /****************************************************************/
 /****************************************************************/
 /*             DFSMSHSM SYSTEM SPECIFICATIONS                   */
 /****************************************************************/
 /*                                                              */
```

```
AUTH ?CTLAUTH               /* ESTABLISH THE USER ID THAT CAN    */ -
 DATABASEAUTHORITY(CONTROL) /* ISSUE AUTH COMMANDS.              */
/*****************************************************************/
/* NOTE: BY DEFAULT, JES3 SUPPORT IS NOT ENABLED FOR DFSMShsm    */
/* HOSTS DEFINED USING HOSTMODE=AUX. CONTACT IBM SUPPORT IF YOU  */
/* REQUIRE JES3 SUPPORT FOR AUX DFSMShsm HOSTS. WHEN JES3 FOR    */
/* DFSMShsm HOSTS IS ENABLED, YOU SHOULD START THE MAIN DFSMShsm */
/* HOST BEFORE STARTING ANY AUX HOSTS AND STOP ALL AUX HOSTS     */
/* BEFORE STOPPING THE MAIN HOST.                                */
/*****************************************************************/

SETSYS ?JESVER              /* JOB ENTRY SUBSYSTEM ID.           */

SETSYS                      /* DO NOT REBLOCK DATA SETS DURING   */ -
  NOCONVERSION              /* RECALL OR RECOVERY.               */

SETSYS                      /* DO NOT ASK OPERATOR PERMISSION TO */ -
  NOREQUEST                 /* START AUTOMATIC FUNCTIONS         */

SETSYS                      /* MOVE OR DELETE DATA WHEN          */ -
  NODEBUG                   /* PERFORMING AUTO FUNCTIONS.        */

SETSYS NOSWAP               /* RUN DFSMSHSM NONSWAPPABLE.        */

SETSYS                      /* DFSMSHSM USES IT OWN FACILITIES   */ -
  DFHSMDATASETSERIALIZATION /* TO SERIALIZE DATA SETS.           */

SETSYS                      /* DFSMSHSM USES ITS DEFINED OPTIMUM */ -
  OPTIMUMDASDBLOCKING       /* BLOCK SIZE WHEN MOVING DATA TO    */
                            /* DFSMSHSM-OWNED DASD.              */
SETSYS                      /* DO NOT USE CMS OF ML1,ML2 AND     */ -
  USECYLINDERMANAGEDSPACE(N) /* BACKUP EAVS                      */
/*****************************************************************/
/*                   DFSMSHSM EXITS                              */
/*****************************************************************/
/*                   NONE ACTIVATED                              */
/*****************************************************************/
/*  DFSMSHSM LOGGING, JOURNALING, AND REPORTING OPTIONS          */
/*****************************************************************/
/*                                                               */

SETSYS                      /* WRITE CDS CHANGES TO JOURNAL      */ -
  JOURNAL(RECOVERY)         /* IMMEDIATELY.                      */

SETSYS                      /* WRITE DAILY STATISTICS RECORDS AND*/ -
  SMF(240)                  /* VOLUME STATISTIC RECORDS TO SMF   */
                            /* RECORD TYPE 240; WRITE FUNCTIONAL */
                            /* STATISTIC RECORDS TO TYPE 241.    */

SETSYS                      /* WRITE ONE COPY OF SYSOUT TO       */ -
  SYSOUT(A 1)               /* PRINTER CLASS A                   */

SETSYS                      /* WRITE DFSMSHSM DUMPS TO SYSTEM    */ -
  SYS1DUMP                  /* DUMP DATA SET.                    */

SETSYS                      /* LOG ALL POSSIBLE DFSMSHSM ACTIVITY*/ -
  ACTLOGMSGLVL(FULL)

SETSYS                      /* WRITE ACTIVITY LOG INFORMATION TO */ -
  ACTLOGTYPE(SYSOUT)        /* THE SYSOUT CLASS SPECIFIED BY THE */
                            /* SYSOUT PARAMETER.                 */
/*                                                               */
/*****************************************************************/
/*                   DFSMSHSM MONITOR OPTIONS                    */
/*****************************************************************/
/*                                                               */
                            /* SPECIFY WHICH INFORMATIONAL       */
```

```
                            /* MESSAGES TO SEND TO THE OPERATOR  */
                            /* CONSOLE.                          */

SETSYS                      /* LIST STARTUP PARAMETERS. DO NOT   */ -
  MONITOR(STARTUP NOVOLUME) /* SEND DATA SET LEVEL MESSAGES TO   */
                            /* THE SYSTEM CONSOLE.               */

SETSYS                      /* DO NOT PRINT SPACE USAGE MSGS.    */ -
  MONITOR(NOSPACE           /* WARN WHEN JOURNAL IS 80% FULL     */ -
        JOURNAL(80))        /*                                   */

/*                                                               */
/*****************************************************************/
/*         DFSMSHSM COMMON SERVICE AREA LIMITING OPTIONS         */
/* THE FOLLOWING CSALIMITS PARAMETERS ARE IGNORED IF             */
/* HOSTMODE=AUX HAS BEEN SPECIFIED AND WILL GENERATE AN ARC0103I */
/* MESSAGE IF ISSUED.                                            */
/*****************************************************************/
/*                                                               */

SETSYS                      /* LIMIT DFSMSHSM'S USAGE OF COMMON  */ -
  CSALIMITS(MWE(4))         /* SEVICE AREA STORAGE. KEEP A       */
                            /* MAXIMUM OF 4 NOWAIT TYPE MWES PER */
                            /* ADDRESS SPACE ON THE CSA QUEUE.   */

SETSYS                      /* NEVER ALLOCATE MORE THEN 100K OF  */ -
  CSALIMITS(MAXIMUM(100)    /* STORAGE FOR MWES. ALLOCATE 90% OF */ -
            ACTIVE(90)      /* AVAILABLE STORAGE TO MWES WHEN    */ -
            INACTIVE(30))   /* DFSMSHSM IS ACTIVE. ALLOCATE ONLY */
                            /* 30% OF AVAILABLE STORAGE WHEN     */
                            /* DFSMSHSM IS INACTIVE.             */
/*                                                               */
/*****************************************************************/
/*               DFSMSHSM TAPE HANDLING SPECIFICATIONS           */
/*****************************************************************/
/*                                                               */

SETSYS                      /* SPECIFY IF EXTENDED TTOCS         */ -
  EXTENDEDTTOC(N)           /* ARE IN USE                        */

SETSYS                      /* USE IMPROVED DATA RECORDING       */ -
  TAPEHARDWARECOMPACT       /* CAPABILITY WHEN 3480X OR NEWER    */
                            /* THE OUTPUT DEVICE.                */

SETSYS                      /* REUSE TAPES THAT ARE PARTIALLY    */ -
  PARTIALTAPE(              /* FULL. DO NOT MARK THEM AS FULL.   */ -
   BACKUP(REUSE) -
   MIGRATION(REUSE))

SETSYS                         /* DO NOT SUSPEND SYSTEM ACTIVITY */ -
  INPUTTAPEALLOCATION(NOWAIT)  /* (WAIT) WHILE INPUT, OUTPUT, OR */ -
  OUTPUTTAPEALLOCATION(NOWAIT) /* RECYCLE TAPES ARE BEING        */ -
  RECYCLETAPEALLOCATION(NOWAIT) /* ALLOCATED.                    */

SETSYS                      /* SCRATCH TAPE SELECTION AT TAPE END*/ -
  SELECTVOLUME(             /* OF VOLUME (EOV) IF FROM THE GLOBAL*/ -
  BACKUP(SCRATCH)           /* IS FROM THE GLOBAL SCRATCH POOL.  */ -
  MIGRATION(SCRATCH) -
  DUMP(SCRATCH))

SETSYS                      /* INFORM THE STORAGE ADMINISTRATOR  */ -
  RECYCLEPERCENT(20)        /* THAT A BACKUP OR MIGRATION TAPE   */
                            /* SHOULD BE RECYCLED WHEN THE AMOUNT*/
                            /* OF TAPE THAT IS OCCUPIED BY VALID */
                            /* DATA IS 20% OR LESS.              */
/*****************************************************************/
/*  IF USERUNITTABLE IS SPECIFIED, IT SHOULD BE CODED PRIOR TO   */
```

```
/*  ASSIGNMENT OF ANY OTHER UNIT STATEMENT.                       */
/****************************************************************/

SETSYS                      /* NO ESOTERIC TAPE DEVICE NAMES ARE */ -
  NOUSERUNITTABLE           /* DEFINED TO DFSMSHSM.              */

SETSYS                      /* UTILIZE 97% OF TAPE CARTRIDGE     */ -
  TAPEUTILIZATION(                                                  -
   UNITTYPE(3590-1) PERCENTFUL(97))

SETSYS                      /* THE AMOUNT OF SPACE THAT MAY NOT  */ -
  TAPESPANSIZE(100)         /* BE UTILIZED AT THE LOGICAL END OF */
                            /* A TAPE CARTRIDGE.                 */

SETSYS                      /* RETURN TAPES THAT NO LONGER       */ -
  TAPEDELETION(             /* CONTAIN VALID DATA TO THE         */ -
   BACKUP(SCRATCHTAPE)      /* GLOBAL SCRATCH POOL.              */ -
   MIGRATION(SCRATCHTAPE) -
   DUMP(SCRATCHTAPE))

SETSYS                      /* WAIT TEN MINUTES BEFORE REISSUING */ -
  MOUNTWAITTIME(10)         /* ADDITIONAL MESSAGES TO TAPE       */
                            /* OPERATORS FOR TAPE MOUNTS.        */

SETSYS                      /* DIREST DFSMSHSM TO INITIALLY      */ -
  UNITNAME(3590-1)          /* SPECIFY A 3590-1 DEVICE FOR       */
                            /* BACKUP OR DUMP SCRATCH TAPES.     */

SETSYS                      /* TAPE OPERATOR MESSAGES            */ -
  TAPEINPUTPROMPT(MIGRATIONTAPES(YES))

SETSYS                      /* TAPE OPERATOR MESSAGES            */ -
  TAPEINPUTPROMPT(BACKUPTAPES(YES))

SETSYS                      /* TAPE OPERATOR MESSAGES            */ -
  TAPEINPUTPROMPT(DUMPTAPES(YES))

SETSYS                      /* TURN ON TAPE DUPLEXING FOR BACKUP */ -
  DUPLEX(                   /* AND MIGRATION. DURING MIGRATION   */ -
   BACKUP(Y)                /* DUPLEXING IF ERRORS ARE           */ -
   MIGRATION(               /* ENCOUNTERED ON THE ALTERNATE TAPE */ -
    Y ERRORALTERNATE(       /* THEN PROCESSING OF THE ORIGINAL   */ -
     CONTINUE)))            /* WILL CONTINUE.                    */

                            /* NUMBER OF ML2 PARTIALS LEFT AFTER */
SETSYS                      /* RECYCLE                           */ -
  ML2PARTIALSNOTASSOCIATEDGOAL(10)

/*                                                              */
/****************************************************************/
/*           DFSMSHSM CONTROL DATA SET BACKUP PARAMETERS        */
/****************************************************************/
/*                                                              */

SETSYS                      /* MAINTAIN FOUR BACKUP VERSIONS     */ -
  CDSVERSIONBACKUP(         /* OF THE CONTROL DATA SETS. BACK    */ -
     BACKUPCOPIES(4)        /* UP THE CONTROL DATA SETS TO       */ -
     BACKUPDEVICECATEGORY(  /* 3590-1 DEVICES IN PARALLEL USING  */ -
       TAPE(UNITNAME(3590-1) /* USING DSS AS THE DATAMOVER       */ -
       PARALLEL))  -
     DATAMOVER(DSS))
/*                                                              */
/****************************************************************/
/*              DFSMSHSM RACF SPECIFICATIONS                    */
/****************************************************************/
/*                                                              */
```

```
SETSYS                      /* DO NOT PUT RACF-INDICATION        */ -
  NORACFIND                 /* ON BACKUP AND MIGRATION           */
                            /* COPIES OF DATA SETS.              */

SETSYS                      /* USE RACF TO PROVIDE TAPE          */ -
  TAPESECURITY(RACF)        /* SECURITY.                         */

SETSYS                      /* DO NOT ALLOW ERASE-ON-SCRATCH     */ -
  NOERASEONSCRATCH          /* ON ANY DFSMSHSM BACKUP            */
                            /* VERSIONS AND MIGRATION COPIES     */

SETSYS                      /* BACKUP DISCRETE RACF PROFILES     */ -
  PROFILEBACKUP

/*                                                               */
/*****************************************************************/
/*                DFSMSHSM COMPACTION OPTIONS                    */
/*****************************************************************/
/*                                                               */

SETSYS                      /* COMPACT DATA SETS THAT MIGRATE TO */ -
  COMPACT(DASDMIGRATE)      /* DASD.                             */

SETSYS                      /* DO NOT COMPACT DATA UNLESS A      */ -
    COMPACTPERCENT(20)      /* SAVINGS OF 20% OR MORE CAN BE     */
                            /* GAINED.                           */

SETSYS -
  OBJECTNAMES(OBJ,OBJECT,LOAD,LOADLIB,LOADMODS,LINKLIB) -
  SOURCENAMES(ASM,COBOL,FORT,PLI,SOURCE,SRC,SRCLIB,SRCE,CNTL,JCL)
/*                                                               */
/*****************************************************************/
/*              DFSMSHSM MIGRATION PARAMETERS                    */
/*****************************************************************/
/*                                                               */

SETSYS                      /* DO NOT ALLOW DFSMSHSM TO MIGRATE  */ -
  TAPEMIGRATION(NONE)       /* DATA SETS TO LEVEL 2 TAPE VOLUMES.*/

SETSYS                      /* SPECIFY PROCESSING WINDOW FOR     */ -
  PRIMARYSPMGMTSTART        /* PRIMARY SPACE MANAGEMENT (LEVEL 0 */ -
  (0000 0000)               /* TO LEVEL 1 MIGRATION              */

DEFINE                      /* RUN PRIMARY SPACE MGMT EVERY      */ -
  PRIMARYSPMGMTCYCLE        /* DAY, STARTING MARCH 02, 1998      */ -
  (YYYYYYY -
  CYCLESTARTDATE(1998/03/02)) SETSYS
                            /* A DATA SET THAT HAS NOT BEEN      */ -
  DAYS(10)                  /* REFERRED TO (OPENED) FOR 10 DAYS  */
                            /* IS ELIGIBLE FOR MIGRATION         */

SETSYS                      /* SPECIFY A HIGH-LEVEL QUALIFIER    */ -
  MIGRATEPREFIX(?UID)       /* WITH WHICH DFSMSHSM RENAMES       */
                            /* MIGRATED DATA SETS.               */

SETSYS                      /* PERFORM INTERVAL MIGRATION        */ -
  INTERVALMIGRATION         /* THROUGHOUT THE DAY.               */

SETSYS                      /* PERFORM ON-DEMAND MIGRATION ON    */ -
  ONDEMANDMIGRATION(Y)      /* SMS VOLUMES IN STORAGE GROUPS     */
                            /* WITH THE ATTRIBUTE AUTOMIGRATE=Y  */

SETSYS                      /* SPECIFY NOTIFICATION LIMIT FOR    */ -
  ODMNOTIFICATIONLIMIT(250) /* ON-DEMAND MIGRATION (250)         */

SETSYS                       /* SPECIFY PROCESSING WINDOW FOR    */ -
  SECONDARYSPMGMTSTART(0000) /* SECONDARY SPACE MANAGEMENT       */
```

```
                             /* (LEVEL 1 TO LEVEL 2 MIGRATION)   */

DEFINE                      /* RUN SECONDARY SPACE MANAGEMENT    */ -
  SECONDARYSPMGMTCYCLE      /* EVERY DAY,                        */ -
  (YYYYYYY                  /* STARTING MARCH 02, 1998.          */ -
  CYCLESTARTDATE(1998/03/02))

SETSYS                      /* KEEP MCDS RECORDS FOR RECALLED    */ -
  MIGRATIONCLEANUPDAYS(10 30 3)  /* DATA SETS FOR 10 DAYS. KEEP  */
                            /* VOLUME OR DAILY STATISTICS RECORDS*/
                            /* FOR 30 DAYS. KEEP RECORDS TO      */
                            /* RECONNECTABLE DATA SETS 3 DAYS    */
                            /* BEYOND EARLIEST ELIGIBILITY.      */

SETSYS                      /* MIGRATE DATA SETS FROM LEVEL 1    */ -
  MIGRATIONLEVEL1DAYS(45)   /* VOLUMES TO LEVEL 2 VOLUMES IF THE */
                            /* DATA SETS HAVE NOT BEEN REFERRED  */
                            /* TO FOR 45 DAYS.                   */

SETSYS                      /* DATA SET EXTENT REDUCTION         */ -
  MAXEXTENTS(10)            /* OCCURS WHEN EXTENTS REACH 10.     */

SETSYS                      /* LIMIT THE NUMBER OF CONCURRENT    */ -
  MAXRECALLTASKS(8)         /* DFSMSHSM RECALL TASKS TO EIGHT.   */


SETSYS                      /* DIRECT DFSMSHSM TO RECALL DATA    */ -
  RECALL(PRIVATEVOLUME(LIKE))/* SETS TO ONLINE VOLUMES WITH THE  */
                            /* USE ATTRIBUTE OF PUBLIC, STORAGE, */
                            /* OR PRIVATE AND WITH LIKE          */
                            /* CHARACTERISTICS.                  */

SETSYS                      /* RETAIN LIST DATA SETS FOR 7 DAYS. */ -
  SCRATCHFREQUENCY(7)       /* DO NOT SCRATCH EXPIRED DATA SETS. */ -
  EXPIREDDATASETS(NOSCRATCH)

SETSYS                      /* DO NOT MIGRATE SMALL DATA SETS AS */ -
  NOSMALLDATASETPACKING     /* RECORDS TO SMALL DATA SET PACKING */
                            /* (SDSP) DATA SETS.                 */

SETSYS                      /* LIMIT THE NUMBER OF CONCURRENT    */ -
  MAXMIGRATIONTASKS(3)      /* AUTOMATIC VOLUME MIGRATION TASKS  */
                            /* TO THREE.                         */

SETSYS                      /* LIMIT THE NUMBER OF CONCURRENT    */ -
  MAXSSMTASKS               /* AUTOMATIC SECONDARY SPACE         */ -
    (CLEANUP(2)             /* MANAGEMENT CLEANUP TASKS TO TWO   */ -
     TAPEMOVEMENT(1))       /* AND TAPEMOVEMENT TASKS TO ONE     */
/*                                                               */
/*****************************************************************/
/*               DFSMSHSM BACKUP PARAMETERS                      */
/*****************************************************************/
/*                                                               */

ONLYIF                      /* THE FOLLOWING DEFINE COMMAND WILL */ -
  HSMHOST(?HOSTID)          /* EXECUTE ONLY IF THE ACTIVE HOST ID*/ -
                            /* MATCHES THE HOST SPECIFIED.       */
                                                                    -
DEFINE BACKUP(Y 1           /* DIRECT DFSMSHSM TO BACKUP ELIGIBLE*/ -
  CYCLESTARTDATE(1998/03/02)) /* DATA SETS DAILY (A 1 DAY CYCLE) */
                            /* TO A SINGLE BACKUP VOLUME,STARTING*/
                            /* MARCH 02, 1998.                   */

SETSYS DSBACKUP(DASDSELECTIONSIZE(3000 250)  DASD(TASKS(2))         -
  TAPE(TASKS(2) DEMOUNTDELAY(MINUTES(60)  MAXIDLETASKS(0))))
                            /* BALANCE THE WORKLOAD BETWEEN TAPE */
                            /* AND DASD FOR WAIT TYPE BACKDS     */
```

```
                            /* COMMANDS. LIMIT THE NUMBER OF DATA*/
                            /* SET BACKUP TAPE AND DASD TASKS.   */
                            /* LIMIT THE NUMBER AND LENGTH OF    */
                            /* TIME A TAPE TASK CAN REMAIN IDLE  */
                            /* BEFORE BEING DEMOUNTED.           */

SETSYS                      /* ACTIVATE THE BACKUP AND DUMP      */ -
  BACKUP                    /* FUNCTION OF DFSMSHSM              */

ONLYIF                      /* THE FOLLOWING SETSYS COMMAND WILL */ -
  HSMHOST(?HOSTID)          /* EXECUTE ONLY IF THE ACTIVE HOST ID*/ -
                            /* MATCHES THE HOST SPECIFIED.       */

SETSYS -
  AUTOBACKUPSTART(0000 0000 0000)
                            /* SPECIFY THE TIME FOR AUTOMATIC    */
                            /* BACKUP TO BEGIN, THE LATEST START */
                            /* TIME THAT AUTOMATIC BACKUP CAN    */
                            /* BEGIN, AND THE QUIESCE TIME FOR   */
                            /* AUTOMATIC BACKUP. NO AUTOMATIC    */
                            /* BACKUP OCCURS UNTIL THESE TIMES   */
                            /* ARE SPECIFIED.                    */

SETSYS                      /* SPECIFY A HIGH-LEVEL QUALIFIER    */ -
  BACKUPPREFIX(?UID)        /* WITH WHICH DFSMSHSM RENAMES BACKED*/
                            /* UP DATA SETS                      */

SETSYS                      /* KEEP ONE VERSION OF EACH BACKED UP*/ -
  VERSIONS(1)               /* DATA SET.                         */ -
  FREQUENCY(0)

SETSYS                      /* LIMIT THE NUMBER OF CONCURRENT    */ -
  MAXBACKUPTASKS(3)         /* BACKUP TASKS TO THREE, BACK UP ALL*/ -
  NOSKIPABPRIMARY           /* DFSMSHSM-MANAGED VOLUMES THAT HAVE*/
                            /* THE AUTO BACKUP ATTRIBUTE.        */

SETSYS                      /* LIMIT THE NUMBER OF CONCURRENT    */ -
  MAXDSRECOVERTASKS(3)      /* DFSMSHSM DATA SET RECOVER TASKS   */
                            /* TO THREE                          */

SETSYS                      /* DURING DAILY BACKUP, MOVE         */ -
  SPILL                     /* DATA SETS FROM FULL DAILY         */
                            /* DASD VOLUMES TO SPILL VOLUMES.    */

SETSYS                      /* MAKE INITIAL BACKUP COPIES OF DATA*/ -
  INCREMENTALBACKUP(ORIGINAL) /* SETS DESPITE THE SETTING OF THE */
                            /* CHANGE BIT.                       */

SETSYS                      /* CREATE DEFAULT VSAM COMPONENT     */ -
   DSBACKUP                 /* NAMES WHEN PROCESSING AN (H)BACKDS*/ -
   (GENVSAMCOMPNAMES(YES))  /* COMMAND AND THE DATA SET BEING    */
                            /* BACKED UP IS VSAM AND THE NEWNAME */
                            /* DATA SET IS UNCATALOGED OR        */
                            /* MIGRATED                          */
/*                                                               */
/*****************************************************************/
/*            DFSMSHSM FULL VOLUME DUMP PARAMETERS               */
/*****************************************************************/
/*                                                               */

ONLYIF                      /* THE DEFINE COMMAND WILL EXECUTE IF*/ -
  HSMHOST(?HOSTID)          /* THE ACTIVE HOST ID = ?HOSTID      */
DEFINE -
  DUMPCYCLE(NNNNNNY         /* 7-DAY DUMP CYCLE WITH DUMP DONE   */ -
   CYCLESTARTDATE(1998/03/02)) /* ONLY ON THE SEVENTH DAY,       */
                            /* STARTING ON MONDAY MARCH 02, 1998,*/
                            /* SO DUMPS OCCUR ON SUNDAY.         */
```

```
 DEFINE DUMPCLASS(SUNDAY DAY(7) -
   RETPD(27) AUTOREUSE NORESET  -
   DATASETRESTORE VTOCCOPIES(4))
                            /* DEFINE A DUMP CLASS NAMED SUNDAY  */
                            /* THAT IS AUTOMATICALLY DUMPED ON   */
                            /* THE SEVENTH DAY OF THE CYCLE.     */
                            /* EACH DUMP COPY IS HELD FOR 27 DAYS*/
                            /* AND THE TAPE IS REUSED WHEN IT IS */
                            /* SCRATCHED. DO NOT RESET DATA SET  */
                            /* CHANGE BITS. ALLOW TSO USERS TO   */
                            /* RESTORE DATA SETS FROM DUMP TAPE. */
                            /* AT MOST, KEEP FOUR VTOC COPY DUMP */
                            /* DATA SETS FOR EACH VOLUME.        */

 DEFINE DUMPCLASS(QUARTERS  /* DEFINE A DUMP CLASS NAMED QUARTERS*/ -
   FREQUENCY(90) RETPD(356) /* THAT IS AUTOMATICALLY DUMPED EVERY*/ -
   NOAUTOREUSE              /* THREE MONTHS AND IS HELD FOR ONE  */ -
   NODATASETRESTORE NORESET /* WEEK LESS THEN A YEAR. USE IS FOR */ -
   DISPOSITION('OFF-SITE')  /* ONLY FULL RESTORES. HOLD THE TAPE */ -
   VTOCCOPIES(0))           /* OFF-SITE AND KEEP NO VTOC COPIES  */
                            /* FOR THIS CLASS.                   */

 SETSYS -
   AUTODUMPSTART(0000 0000 0000)
                            /* SPECIFY THE TIME FOR AUTOMATIC    */
                            /* DUMP TO BEGIN, THE LATEST START   */
                            /* THAT AUTOMATIC DUMP CAN BEGIN,    */
                            /* AND THE QUIESCE TIME FOR AUTOMATIC*/
                            /* DUMP. NO AUTOMATIC DUMP OCCURS    */
                            /* UNTIL THESE TIMES ARE SPECIFIED.  */

 SETSYS                     /* BUFFER FIVE TRACKS WHEN PERFORMING*/ -
   DUMPIO(3,2)              /* A DUMP. BUFFER TWO TRACKS DURING  */
                            /* DATA MOVEMENT.                    */

 SETSYS                     /* LIMIT THE NUMBER OF CONCURRENT    */ -
   MAXDUMPTASKS(3)          /* DUMP TASK TO THREE.               */
 /*                                                              */
 /****************************************************************/
 /*      DFSMSHSM AGGREGATE BACKUP AND RECOVER PARAMETERS        */
 /* THE FOLLOWING ABARS PARAMETERS ARE IGNORED IF HOSTMODE=AUX   */
 /* HAS BEEN SPECIFIED AND WILL GENERATE AN ARC0103I MESSAGE IF  */
 /* ISSUED.                                                      */
 /****************************************************************/
 /*                                                              */

SETSYS                      /* RECOVER DATA SET AGGREGATES TO    */ -
  ARECOVERUNITNAME(3590-1)  /* 3590-1 TAPE DEVICES.              */

SETSYS                      /* START ONLY ONE SECONDARY ADDRESS  */ -
  MAXABARSADDRESSSPACE(1)   /* SPACE FOR BACKING UP AND          */
                            /* RECOVERING AGGREGATED DATA SETS   */

SETSYS                      /* START THE SECONDARY ADDRESS       */ -
  ABARSPROCNAME(DFHSMABR)   /* SPACE WITH THE STARTUP PROCEDURE  */
                            /* NAMED DFHSMABR.                   */

SETSYS                      /* WRITE THE ABARS ACTIVITY LOG TO   */ -
  ABARSACTLOGTYPE(DASD)     /* DASD                              */

SETSYS                      /* LOG ALL ABARS MESSAGES            */ -
  ABARSACTLOGMSGLVL(FULL)

SETSYS                      /* RECOVER ML2 DATA SETS TO TAPE.    */ -
  ARECOVERML2UNIT(3590-1)
```

```
SETSYS                        /* USE 90% OF THE AVAILABLE TAPE FOR */ -
  ARECOVERPERCENTUTILIZED(090) /* ARECOVERY TAPES.                 */

SETSYS                        /* BACKUP AGGREGATES TO 3590-1       */ -
  ABARSUNITNAME(3590-1)       /* DEVICES.

SETSYS                        /* BACKUP ABARS DATA SETS WITH TWO   */ -
  ABARSBUFFERS(2)             /* DATA MOVEMENT BUFFERS.            */

SETSYS                        /* SPECIFY ABARS TO STACK THE        */ -
  ABARSTAPES(STACK)           /* ABACKUP OUTPUT ONTO A MINIMUM     */
                              /* NUMBER OF TAPE VOLUMES            */

SETSYS                        /* ABARS ACTIVITY LOG WILL NOT BE    */ -
  ABARSDELETEACTIVITY(N)      /* AUTOMATICALLY DELETED DURING      */
                              /* ABARS PROCESSING                  */

SETSYS                        /* SET PERFORMANCE OF BACKING UP     */ -
  ABARSOPTIMIZE(3)            /* LEVEL 0 DASD DATASETS             */

SETSYS                        /* TARGET DATASET IS TO BE ASSIGNED  */ -
  ARECOVERTGTGDS(SOURCE)      /* SOURCE STATUS                     */

SETSYS                        /* ALLOWS RECOVERY OF A LEVEL 0      */ -
  ABARSVOLCOUNT(ANY)          /* DASD DATA SET UP TO 59 VOLUMES    */
 /*                                                                */
 /****************************************************************/
 /*            DFSMSHSM HSMPLEX/SYSPLEX PARAMETERS                 */
 /****************************************************************/
 /*                                                                */

SETSYS                        /* SPECIFY THE SUFFIX FOR THE        */ -
  PLEXNAME(PLEX0)             /* HSMPLEX IN A MULTI-HSMPLEX        */
                              /* ENVIRONMENT   ARC(SUFFIX)         */

SETSYS                        /* SPECIFY HOST NOT TO TAKE OVER     */ -
  PROMOTE(PRIMARYHOST(NO)     /* PRIMARY OR SSM RESPONSIBILITIES   */ -
  SSM(NO))
/*                                                                 */
/*****************************************************************/
/* YOU MAY REMOVE ADDVOL COMMANDS FOR VOLUMES OTHER THAN PRIMARY */
/* AND MIGRATION LEVEL 1 VOLUMES FROM ARCCMD__ IF YOU WANT TO    */
/* SAVE TIME DURING DFSMSHSM STARTUP.  THOSE ADDVOL COMMANDS ARE */
/* STORED IN THE CONTROL DATA SETS WHEN DFSMSHSM IS STARTED.     */
/*****************************************************************/
/*                                                                 */
ADDVOL ______               /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)         /* WITH UNIT TYPE (PROVIDE TYPE)  */ -
       BACKUP               /* AS A DAILY BACKUP VOLUME FOR   */ -
       (DAILY)              /* AUTOMATIC BACKUP.              */ -
       THRESH(97)           /* SPILL CONTENTS UNTIL THIS      */ -
                            /* VOLUME IS 97% FULL.            */

ADDVOL ______               /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)         /* WITH UNIT TYPE (PROVIDE TYPE)  */ -
       BACKUP               /* AS A SPILL BACKUP VOLUME THAT  */ -
       (SPILL)              /* IS CONSIDERED FULL AND         */ -
       THRESH(97)           /* UNUSABLE WHEN 97% FULL.        */

ADDVOL ______               /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)         /* THAT IS A TAPE USED AS         */ -
       BACKUP(DAILY)        /* A DAILY BACKUP VOLUME.         */

ADDVOL ______               /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       DUMP                 /* USED FOR FULL VOLUME           */ -
       (DUMPCLASS(SUNDAY))  /* DUMP FOR SUNDAY CLASS.         */ -
       UNIT(______)         /* DUMPS MUST GO TO TAPE.         */
```

```
 /*                                                             */
./ ADD NAME=ARCCMD01
 /******************************************************************/
 /*  DFSMSHSM STARTUP COMMAND MEMBER FOR LEVEL 2 TAPE MIGRATION  */
 /*                                                                */
 /*  APPEND THIS COMMAND STREAM TO ARCCMD00 TO PROVIDE LEVEL 2   */
 /*  TAPE MIGRATION                                                */
 /******************************************************************/
 /******************************************************************/
 /*         DFSMSHSM LEVEL 2 TAPE MIGRATION PARAMETERS           */
 /******************************************************************/
 /*                                                                */

 SETSYS -
   TAPEMIGRATION(ML2TAPE)    /* MIGRATE TO LEVEL 2 TAPE.          */

 SETSYS -
   MIGUNITNAME(3590-1)       /* START WITH 3590-1 ML2 TAPE        */
                             /* UNIT.                             */
 SETSYS -
   ML2RECYCLEPERCENT(20)     /* LOG MESSAGE WHEN VALID DATA       */
                             /* ON AN ML2 TAPE FALLS BELOW        */
                             /* 20%.                              */
 SETSYS -
   TAPEMAXRECALLTASKS(1)     /* ONE TAPE RECALL TASK AT A TIME    */

 /*                                                                */
 /******************************************************************/
 /* SEE MEMBER ARCCMD91 IN HSM.SAMPLE.CNTL FOR AN EXAMPLE         */
 /* OF ADDVOL COMMANDS TO BE USED IN CONJUNCTION WITH LEVEL       */
 /* 2 TAPE MIGRATION.                                              */
 /******************************************************************/
 /*                                                                */
```

Adapting and using the starter set

The following members provide examples for other DFSMShsm management functions. With initial planning, you can use each member or job to perform tasks in your environment. (For information about planning the DFSMShsm environment, see *z/OS Migration*.) As you gradually implement the following members or jobs, you might want to review Part 2, “Customizing DFSMShsm,” on page 277.

Member	Description
“ARCCMD90” on page 125	A sample member that uses ADDVOL commands to define a primary and a migration level 1 volume to DFSMShsm. You can also optionally use this member to define daily backup, spill, and dump volumes as needed. After you have edited this member, append it to the ARCCMD00 member in the SYS1.PARMLIB data set. If you want DFSMShsm to process SMS-managed volumes automatically, assign those volumes to storage groups that have the automatic function attribute of YES.

Member	Description
"ARCCMD01" on page 126 and "ARCCMD91" on page 127	These are members that provide examples of how to identify a migration level 2 tape to DFSMShsm. The structure illustrated in these members is the way you define other tapes to DFSMShsm. ARCCMD91 provides an example ADDVOL command that defines migration level 2 tape devices to DFSMShsm. ARCCMD01 provides example SETSYS commands that define the migration level 2 tape processing environment to DFSMShsm. After you have edited these members, append them to the ARCCMD00 member in the SYS1.PARMLIB data set.
"HSMHELP" on page 128	This member provides help text about DFSMShsm-authorized commands for users with data base authority. A copy should be placed where users with database authority can invoke the help text.
"HSMLOG" on page 141	This member provides a sample job to print the DFSMShsm log.
"HSMEDIT" on page 142	This member provides a sample job to print the edit log.
"ALLOCBK1" on page 142	This member provides a sample job that defines four backup versions of each control data set and allocates those backup versions on DASD volumes (tapes can also be selected). **Note:** Four backup versions of each control data set are the default, but this number can be changed with the BACKUPCOPIES parameter of the SETSYS command.
"ALLOSDSP" on page 145	This member provides a sample job that allocates a small-data-set-packing data set.
"HSMPRESS" on page 147	This member provides a sample job that reorganizes the control data sets.

ARCCMD90

ARCCMD90 helps you configure your DFSMShsm environment with primary, migration level 1, daily backup, spill, and dump volumes. Refer to Figure 28 on page 126 for an example listing of the ARCCMD90 member.

You need to specify the primary and migration level 1 volumes as follows:

- Primary volumes: Use TSO and batch volumes containing non-VSAM data sets and VSAM spheres cataloged in an integrated catalog facility catalog. (VSAM data sets not cataloged in an integrated catalog facility catalog are supported only by backup).
- Migration level 1 volumes: These need not be volumes dedicated to DFSMShsm use, but see "User or system data on migration level 1 volumes" on page 95 for considerations if they are not so dedicated. If not dedicated, choose volumes with the most free space (**Example:** Catalog packs, or packs with swap or page data sets).

Tip: You can optionally specify the daily backup, spill backup, and dump backup volumes as tape volumes.

After you have edited ARCCMD90, append it to member ARCCMD00.

```
 /****************************************************************/
 /* THE FOLLOWING COMMANDS ARE AN EXAMPLE OF ADDVOL COMMANDS      */
 /* USED IN CONJUNCTION WITH THE ARCCMD00 COMPONENT OF THE        */
 /* STARTER SET.  THEY CAN BE COMPLETED BY ADDING VOLUME SERIAL   */
 /* NUMBERS AND UNIT TYPES IN THE SPACES PROVIDED.  THIS COMMAND  */
 /* STREAM CAN THEN BE APPENDED TO ARCCMD00.                      */
 /*                                                               */
 /* ADDVOL COMMANDS FOR PRIMARY AND MIGRATION LEVEL 1 VOLUMES     */
 /* MUST BE INCLUDED IN THE ARCCMD__ PARMLIB MEMBER FOR YOUR      */
 /* SYSTEM.  INDEED, THEY MUST BE IN THE ARCCMD__ WHEN RUNNING    */
 /* WITH JES3.                                                    */
 /*                                                               */
 /****************************************************************/
 /*                                                               */
ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL) */ -
       UNIT(______)        /* WITH UNIT TYPE (PROVIDE TYPE) */ -
       PRIMARY             /* AS A PRIMARY VOLUME THAT IS A */ -
       (AUTOMIGRATION      /* CANDIDATE FOR AUTOMIGRATION.  */ -
        MIGRATE(7)         /* MIGRATE DATA SETS AFTER 7 DAYS*/ -
        AUTOBACKUP         /* CANDIDATE FOR AUTOBACKUP.     */ -
        BACKUPDEVICECATEGORY(TAPE) /* BACKED UP TO TAPE.    */ -
        AUTORECALL         /* DATA SETS CAN BE RECALLED TO  */ -
                           /* THIS VOLUME.                  */ -
        AUTODUMP(SUNDAY))  /* DUMP FULL VOLUME AS SPECIFIED */ -
                           /* BY THE SUNDAY DUMP CLASS.     */ -
       THRESHOLD(100 0)    /* NO INTERVAL MIGRATION         */
ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL) */ -
       UNIT(______)        /* WITH UNIT TYPE (PROVIDE TYPE) */ -
       MIGRATION           /* AS A MIGRATION LEVEL 1 VOLUME */ -
       (MIGRATIONLEVEL1    /* WITH NO SMALL DATA SET        */ -
        NOSMALLDATASETPACKING) /* PACKING AVAILABLE.        */ -
       THRESHOLD(100)      /* NO THRESHOLD PROCESSING.      */
 /*                                                               */
 /****************************************************************/
 /* YOU MAY REMOVE ADDVOL COMMANDS FOR VOLUMES OTHER THAN PRIMARY */
 /* AND MIGRATION LEVEL 1 VOLUMES FROM ARCCMD__ IF YOU WANT TO    */
 /* SAVE TIME DURING DFSMSHSM STARTUP.  THOSE ADDVOL COMMANDS ARE */
 /* STORED IN THE CONTROL DATA SETS WHEN DFSMSHSM IS STARTED.     */
 /****************************************************************/
 /*                                                               */
ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)        /* WITH UNIT TYPE (PROVIDE TYPE)  */ -
       BACKUP              /* AS A DAILY BACKUP VOLUME FOR   */ -
       (DAILY)             /* AUTOMATIC BACKUP.              */ -
       THRESH(97)          /* SPILL CONTENTS UNTIL THIS      */ -
                           /* VOLUME IS 97% FULL.            */

ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)        /* WITH UNIT TYPE (PROVIDE TYPE)  */ -
       BACKUP              /* AS A SPILL BACKUP VOLUME THAT  */ -
       (SPILL)             /* IS CONSIDERED FULL AND         */ -
       THRESH(97)          /* UNUSABLE WHEN 97% FULL.        */

ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       UNIT(______)        /* THAT IS A TAPE USED AS         */ -
       BACKUP(DAILY)       /* A DAILY BACKUP VOLUME.         */

ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL)  */ -
       DUMP                /* USED FOR FULL VOLUME           */ -
       (DUMPCLASS(SUNDAY)) /* DUMP FOR SUNDAY CLASS.         */ -
       UNIT(______)        /* DUMPS MUST GO TO TAPE.         */
```

Figure 28. Example Listing of Member ARCCMD90

ARCCMD01

ARCCMD01 helps you configure your DFSMShsm environment using JES2 for migration from primary volumes and migration level 1 volumes to tape migration level 2 volumes. ARCCMD01 contains DFSMShsm parameter specifications. Refer to Figure 29 for an example listing of the ARCCMD01 member.

You need to specify primary volumes, migration level 1 volumes, and tape migration level 2 volumes. See "ARCCMD91" for an example of how to specify the tape volumes with the ADDVOL command.

After you have edited ARCCMD01, append it to member ARCCMD00.

```
 /****************************************************************/
 /*  DFSMSHSM STARTUP COMMAND MEMBER FOR LEVEL 2 TAPE MIGRATION  */
 /*                                                              */
 /*  APPEND THIS COMMAND STREAM TO ARCCMD00 TO PROVIDE LEVEL 2   */
 /*  TAPE MIGRATION                                              */
 /****************************************************************/
 /****************************************************************/
 /*          DFSMSHSM LEVEL 2 TAPE MIGRATION PARAMETERS          */
 /****************************************************************/
 /*                                                              */

SETSYS -
  TAPEMIGRATION(ML2TAPE)    /* MIGRATE TO LEVEL 2 TAPE.         */

SETSYS -
  MIGUNITNAME(3590-1)       /* START WITH 3590-1 ML2 TAPE       */
                            /* UNIT.                            */
SETSYS -
  ML2RECYCLEPERCENT(20)     /* LOG MESSAGE WHEN VALID DATA      */
                            /* ON AN ML2 TAPE FALLS BELOW       */
                            /* 20%.                             */
SETSYS -
  TAPEMAXRECALLTASKS(1)     /* ONE TAPE RECALL TASK AT A TIME   */

 /*                                                              */
 /****************************************************************/
 /* SEE MEMBER ARCCMD91 IN HSM.SAMPLE.CNTL FOR AN EXAMPLE        */
 /* OF ADDVOL COMMANDS TO BE USED IN CONJUNCTION WITH LEVEL      */
 /* 2 TAPE MIGRATION.                                            */
 /****************************************************************/
 /*                                                              */
```

Figure 29. Example Listing of Member ARCCMD01

ARCCMD91

ARCCMD91 helps you configure your DFSMShsm environment using a migration level 2 tape volume. The following member provides a sample ADDVOL command for the DFSMShsm-started procedure. Refer to Figure 30 on page 128 for an example listing of the ARCCMD91 member.

Migration level 2 volumes typically are tape volumes; you can, however, specify DASD volumes.

After editing this member, append the member to ARCCMD00.

```
/*                                                              */
/****************************************************************/
/* THE FOLLOWING EXAMPLE ADDVOL COMMANDS CAN BE USED WITH THE   */
/* ARCCD00 MEMBER OF THE STARTER SET TO IDENTIFY LEVEL 2 TAPE   */
/* MIGRATION VOLUMES.  AFTER YOU HAVE ADDED A VOLUME SERIAL     */
/* NUMBER AND A UNIT TYPE IN THE SPACE PROVIDED, APPEND THIS    */
/* COMMAND STREAM TO YOUR ARCCMD00 MEMBER.                      */
/*                                                              */
/****************************************************************/
/*                                                              */
ADDVOL ______              /* ADD A VOLUME (PROVIDE SERIAL)    */ -
       MIGRATION           /* AS A MIGRATION LEVEL 2 TAPE      */ -
        (MIGRATIONLEVEL2)  /* VOLUME.                          */ -
       UNIT(______)        /* PROVIDE PROPER UNIT TYPE.        */
/*                                                              */
```

Figure 30. Example Listing of Member ARCCMD91

HSMHELP

HSMHELP helps you adapt DFSMShsm to your environment. HSMHELP is a HELP member that explains syntax for operands of the HSENDCMD command. Refer to "Example Listing of Member HSMHELP" for an example listing of the HSMHELP member.

Note: Copy the HSMHELP member into a data set where only users with data base authority can refer to these DFSMShsm commands.

Example Listing of Member HSMHELP

```
)F Function:
             The HSENDCMD command is used by authorized TSO users
             to communicate with the DFSMShsm functions.
)X SYNTAX:
             HSENDCMD(WAIT | NOWAIT) command

REQUIRED  -  command - You must enter a command.
DEFAULTS  -  none
ALIAS     -  none

)O OPERANDS:
            'command' - specifies the DFSMShsm operator command.
DEFAULTS  -  none

The following is a list of all DFSMShsm commands except
the user commands:

ABACKUP - Back up aggregated date sets
ADDVOL - Add or change the volumes to be controlled by DFSMShsm
ALTERDS - Change the backup specifications for a data set
ARECOVER - Recover aggregated date sets
AUDIT - Audit DFSMShsm
AUTH - Authorize a TSO user for DFSMShsm commands
BACKDS - Create a backup version of a data set
BACKVOL - Create a backup version of all data sets on a volume or
          or on CDS
BDELETE - Delete a backup version of a data set
CANCEL - Cancel a queued DFSMShsm request
DEFINE - Define control structures to DFSMShsm
DELETE - Delete a data set that has been migrated
DELVOL - Remove a volume from DFSMShsm control
DISPLAY - Display DFSMShsm storage locations
EXPIREBV - Delete unwanted backup versions of data sets
FIXCDS - Repair a DFSMShsm control data set
```

```
FRBACKUP - Create a fast replication backup of a copy pool
FRDELETE - Delete a backup version of a copy pool
FREEVOL - Move migrated data sets from migration volumes,
          and backup data sets from backup volumes
FRRECOV - Re-create a volume or copy pool from a backup version
HOLD - Suspend a DFSMShsm function
LIST - List information from the DFSMShsm control data sets
LOG - Enter data into the DFSMShsm Log
MIGRATE - Space manage a specific volume or migrate a data set
PATCH - Modify DFSMShsm storage locations
QUERY - List the status of DFSMShsm parameters, statistics, requests
RECALL - Recall a data set
RECOVER - Re-create a data set or a volume from a backup version
RECYCLE - Move valid backup or migration copies from one tape
          to another
RELEASE - Resume a DFSMShsm function
REPORT - Request reports based on DFSMShsm statistics
SETMIG - Change the eligibility for migration of data sets
SETSYS - Define or change the DFSMShsm installation parameters
STOP - Stop the DFSMShsm system task
SWAPLOG - Switch the DFSMShsm log data sets
TAPECOPY - Copy a DFSMShsm-owned migration or backup tape volume
           to an alternate volume
TAPEREPL - Replace a DFSMShsm-owned migration or backup tape volume
           with an alternate volume
TRAP - Request a dump when a specified error occurs
UPDATEC - Apply the DFSMShsm journal to recover a control data set
The following list shows specific information about each command.
You could request the same information by typing HELP HSMHELP
OPERANDS(command).))ABACKUP  agname
           UNIT(unittype)
           EXECUTE | VERIFY
           MOVE
           FILTEROUTPUTDATASET(dsname)
           PROCESSONLY(LEVEL0 | MIGRATIONLEVEL1 | MIGRATIONLEVEL2 |
                       USERTAPE)
           STACK | NOSTACK
           OPTIMIZE(1|2|3|4)
           SKIP(PPRC | XRC | NOPPRC | NOXRC)
           LIST(SKIPPED)
))ADDVOL   volser
           BACKUP | DUMP | MIGRATION | PRIMARY
           UNIT(unittype)
               (AUTOBACKUP | NOAUTOBACKUP)
               (AUTODUMP(class,(class,class,class,class))|NOAUTODUMP)
               (AUTOMIGRATION | NOAUTOMIGRATION)
               (AUTORECALL | NOAUTORECALL)
               (BACKUPDEVICECATEGORY(TAPE | DASD | NONE))
               (DAILY(day) | SPILL)
               (DELETEBYAGE(days) | DELETEIFBACKEDUP(days) |
               MIGRATE(days))
               DENSITY(2|3|4)
               (DRAIN | NODRAIN)
               (OVERFLOW | NOOVERFLOW)
               (DUMPCLASS(class))
               (MIGRATIONLEVEL1 | MIGRATIONLEVEL2)
               (SMALLDATASETPACKING | NOSMALLDATASETPACKING)
               THRESHOLD(thresh1(thresh2))
               TRACKMANAGEDTHRESHOLD(thresh1 thresh2)

))ALTERDS  (dsname...)
                FREQUENCY(days) | SYSFREQUENCY
                VERSIONS(limit) | SYSVERSIONS

))ARECOVER DATASETNAME(controlfiledsname) |
               STACK | NOSTACK
               VOLUMES(volser1 ... volsern) | XMIT
```

```
                UNIT(unittype)
        AGGREGATE(agname)
            DATE(yyyy/mm/dd) | VERSION(nnnn)
        EXECUTE | VERIFY | PREPARE
            ACTIVITY
            DATASETCONFLICT
             (RENAMESOURCE(level) |
              RENAMETARGET(level) |
              BYPASS | REPLACE)
            INSTRUCTION
            MENTITY(modeldsn)
            MIGRATEDDATA(ML1 | ML2 | SOURCELEVEL)
            NOBACKUPMIGRATED
            ONLYDATASET
             (NAME(dsname) |
              LISTOFNAMES(listdsname))
            PERCENTUTILIZED(percent)
            RECOVERNEWNAMEALL(level)|
             RECOVERNEWNAMELEVEL(olevel1,nlevel1, ...,)
            TARGETUNIT(unittype)
            TGTGDS(SOURCE | ACTIVITY | DEFERRED | ROLLEDOFF)
            VOLCOUNT(ANY | NONE)

))AUDIT -  Command Variations:

  AUDIT    ABARCONTROLS | ABARSCONTROLS(agname)  AUDIT    ALL
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class)
                REPORT(ALL | ERRORS)
                SERIALIZATION(CONTINUOUS)

  AUDIT    BACKUPTYPE(DAILY(day) | SPILL | ALL)
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class)
                REPORT(ALL | ERRORS)
                SERIALIZATION(CONTINUOUS)

  AUDIT    BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET |
            OFFLINECONTROLDATASET(DAILY(day) | ML2 | SPILL | ALL)
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
                REPORT(ALL | ERRORS)
                SERIALIZATION(CONTINUOUS)

  AUDIT    BACKUPVOLUMES(volser ...)
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
                REPORT(ALL | ERRORS)
                SERIALIZATION(CONTINUOUS)

  AUDIT    COMMONQUEUE(RECALL)
                FIX | NOFIX

  AUDIT    COPYPOOLCONTROLS
             (cpname)

  AUDIT    DATASETCONTROLS(MIGRATION | BACKUP)
                DATASETNAMES(dsname ...) | LEVELS(qualifier ...) |
                RESUME
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class)
                REPORT(ERRORS)
                SERIALIZATION(DYNAMIC | CONTINUOUS)

  AUDIT    DATASETNAMES(dsname ...)
                FIX | NOFIX
                OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
```

```
         REPORT(ALL)
         SERIALIZATION(CONTINUOUS)

AUDIT    DIRECTORYCONTROLS VOLUMES(volser)
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class)
         REPORT(ERRORS)
         SERIALIZATION(DYNAMIC | CONTINUOUS)

AUDIT    LEVELS(qualifier ...)
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
         REPORT(ALL | ERRORS)
         SERIALIZATION(CONTINUOUS)

AUDIT    MASTERCATALOG | USERCATALOG(catname)
         NOFIX
         OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
         REPORT(ALL | ERRORS)
         SERIALIZATION(CONTINUOUS)

AUDIT    MEDIACONTROLS(SMALLDATASETPACKING)
         VOLUMES(volser)
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class)
         REPORT(ERRORS)
         SERIALIZATION(DYNAMIC | CONTINUOUS)
AUDIT    VOLUMES(volser ...)
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
         REPORT(ALL | ERRORS)
         SERIALIZATION(CONTINUOUS)

AUDIT    VOLUMECONTROLS(BACKUP)
         VOLUMES(volser ...) | BACKUPTYPE(DAILY(day)) |
         SPILL | ALL
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class)
         REPORT(ERRORS)
         SERIALIZATION(DYNAMIC | CONTINUOUS)

AUDIT    VOLUMECONTROLS(MIGRATION | RECOVERABLE)
         VOLUMES(volser ...)
         FIX | NOFIX
         OUTDATASET(dsname) | SYSOUT(class)
         REPORT(ERRORS)
         SERIALIZATION(DYNAMIC | CONTINUOUS)

))AUTH     userid
           DATABASEAUTHORITY(USER | CONTROL) | REVOKE

))BACKDS   dsname
             TARGET(DASD | TAPE)
             NEWNAME(newdsname)
                  DATE(yyyy/mm/dd)
                       TIME(hhmmss)
                  SPHERE(YES | NO)
                  GENVSAMCOMPNAMES(YES | NO)
             CC(PREFERRED | CACHEPREFERRED |
                  VIRTUALPREFERRED | REQUIRED |
                  CACHEREQUIRED | VIRTUALREQUIRED |
                  STANDARD
                  PHYSICALEND | LOGICALEND)
             UNIT(unittype)
             VOLUME(volser)
             RETAINDAYS(days)
```

```
))BACKVOL  PRIMARY | VOLUMES(volser...) | STORAGEGROUP(sgname ...) |
                CONTROLDATASETS(
                     DATAMOVER(HSM | DSS)
                     BACKUPDEVICECATEGORY
                          (DASD | TAPE(PARALLEL | NOPARALLEL)) |
                     NULLJOURNALONLY)
                FREQUENCY(days)
                INCREMENTAL | TOTAL
                DUMP(DUMPCLASS(class,class,class,class,class)
                RETENTIONPERIOD(days | * | NOLIMIT ...)
                STACK(nnn | * ...) )
                TERMINAL
                UNIT(unittype)

))BDELETE  (dsname...)
                ALL | VERSIONS(bvn ...) |
                DATE(yyyy/mm/dd) TIME(hhmmss)
                FROMVOLUME(volser)
))CANCEL        DATASETNAME(dsn) | REQUEST(num) | USERID(userid)

))DEFINE        ARPOOL(agname | ALL
                              | ML1VOLS(* | volser ... volsern)
                              | L0VOLS(* | volser ... volsern))
                BACKUP(cycle(bvol) CYCLESTARTDATE(yyyy/mm/dd))
                DUMPCLASS(class)
                AUTOREUSE | NOAUTOREUSE
                DATASETRESTORE | NODATASETRESTORE
                DAY(day)
                DISABLE
                DISPOSITION('disposition')
                FREQUENCY(days)                RESET | NORESET
                RETENTIONPERIOD(days | NOLIMIT
                STACK(nnn)
                SWITCHTAPES(DSBACKUP(TIME(hhmm) | AUTOBACKUPEND
                                     PARTIALTAPE(REUSE |
                                                 MARKFULL |
                                                 SETSYS)))

                TAPEEXPIRATIONDATE(yyyyddd)
                UNIT(unittype)
                VTOCCOPIES(copies)
                DUMPCYCLE(cycle CYCLESTARTDATE(yyyy/mm/dd))
                MIGRATIONCLEANUPCYCLE(cycle(CYCLESTARTDATE(yyyy/mm/dd)))
                MIGRATIONLEVEL2(KEYS(key ...) VOLUMES(volser...))
                POOL(poolid VOLUMES(volser))
                PRIMARYSPMGMTCYCLE(cycle CYCLESTARTDATE(yyyy/mm/dd))
                SECONDARYSPMGMTCYCLE(cycle CYCLESTARTDATE(yyyy/mm/dd))
                VOLUMEPOOL(poolid VOLUMES(volser))

))DELETE   dsn
                PURGE

))DELVOL   volser
           BACKUP | DUMP | MIGRATION | PRIMARY
               (PURGE | REASSIGN | UNASSIGN | MARKFULL
               LASTCOPY
               COPYPOOLCOPY)

))DISPLAY  (address (:address)...)
           LENGTHS(bytes...)
           LOGONLY
           OUTDATASET(dsname)
           VOLSER(volser)

))EXPIREBV DISPLAY | EXECUTE
                ABARVERSIONS
                ABARVERSIONS(AGNAME(agname))
```

```
                RETAINVERSIONS(n)
                NONSMSVERSIONS(DELETEIFBACKEDUP(days)
                CATALOGEDDATA(days)
                UNCATALOGEDDATA(days))
                STARTKEY(lowkey) | RESUME
                ENDKEY(highkey)
                OUTDATASET(dsname) | SYSOUT(class)

))FIXCDS   type
           key
            ADDMIGRATEDDATASET(volser) | ASSIGNEDBIT(ON | OFF) |
            CREATE(offset data) | DELETE | DISPLAY(offset) |
            EXPAND(bytes) | NEWKEY(keyname) |
            VERIFY(offset data | BITS(bits))|
            PATCH(offset data | BIT(bits))
            ENTRY(volser dsname)
            LENGTH(bytes)
            LOGONLY
            OUTDATASET(dsname)
            REFRESH(ON|OFF)

))FREEVOL MIGRATIONVOLUME(volser)
            AGE(days)
            TARGETLEVEL(MIGRATIONLEVEL1 | MIGRATIONLEVEL2(TAPE|DASD))
          BACKUPVOLUME(volser)
            AGE(days)
            TARGETLEVEL(SPILL (TAPE | DASD))
            RETAINNEWESTVERSION))FRBACKUP COPYPOOL(cpname)
             EXECUTE
               TOKEN(token)
               NOVTOCENQ
               FORCE
               DUMP
                 RETURNCONTROL(DUMPEND | FASTREPLICATIONEND)
                 DUMPCLASS(dclass1,...,dclass5)
             PREPARE
               FORCE
             WITHDRAW
             DUMPONLY(
               TOKEN(token) | VERSION(vernum) | DATE(yyyy/mm/dd) |
               GENERATION(gennum))
               DUMPCLASS(dclass1,...,dclass5)

))FRDELETE COPYPOOL(cpname)
             VERSIONS(ver,...) | TOKEN(token) | ALL
             BOTH | DASDONLY | DUMPONLY(DUMPCLASS(dclass1,...,dclass5))

))FRRECOV  COPYPOOL(cpname)
             FORCE
             VERIFY(Y | N)
             FROMDASD |
             FROMDUMP(
               DUMPCLASS(dclass) | PARTIALOK | RESUME(YES | NO)
             RSA(keylbl)
           DSNAME(dsname , ...)
             REPLACE
             FROMCOPYPOOL(cpname)
             FROMDASD |
             FROMDUMP(
               APPLYINCREMENTAL
               DUMPCLASS(dclass) | DUMPVOLUME(dvol))
               RSA(keylbl)
             FASTREPLICATION(PREFERRED | NONE | REQUIRED)
             NOCOPYPOOLBACKUP(RC8 | RC4)
           TOVOLUME(volser)
             FROMCOPYPOOL(cpname)
             FROMDASD |
```

```
            FROMDUMP(
              APPLYINCREMENTAL
              DUMPCLASS(dclass) | DUMPVOLUME(dvol))
              RSA(keylbl)
          DATE(yyyy/mm/dd) | GENERATION(gennum) |
          TOKEN(token) | VERSION(vernum) |
          ALLOWPPRCP(
            NO | YES |
            PRESERVEMIRRORPREFERRED |
            PRESERVEMIRRORREQUIRED)
))HOLD         ABACKUP(agname)
               ALL
               ARECOVER
                 AGGREGATE(agname) | DATASETNAME(controlfiledsn)
               AUDIT
               AUTOMIGRATION
               BACKUP(AUTO
                      DSCOMMAND(
                                DASD |
                                TAPE |
                                SWITCHTAPES))
               COMMONQUEUE(
                 RECALL(
                        SELECTION | PLACEMENT))
               DUMP(AUTO | FASTREPLICATIONBACKUP)
               ENDOFDATASET | ENDOFVOLUME
               EXPIREBV
               FRBACKUP
               FRRECOV(
                DATASET | TAPE))
               LIST
               LOG                 MIGRATION(AUTO)
               RECALL(TAPE(TSO))
               RECOVER(TAPEDATASET)
               RECYCLE
               REPORT
               TAPECOPY
               TAPEREPL

))HSENDCMD command
               WAIT | NOWAIT

If you are working from the DFSMShsm panel and your command
fits on the COMMAND === line of the panel, then simply type
TSO HSENDCMD ---command---
If you need space for a multiline command, then split the
screen and select 'OPTION 6'.
Type in the multiline command.
After the command has been processed, return to the DFSMShsm panel.

))LIST  -  Command Variations :

  LIST     AGGREGATE(agname)
           AGGREGATE(*)
             DATE(yyy/mm/dd)
             VERSION(nnnn)

  LIST     BACKUPVOLUME(volser)
           OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
           SELECT(EMPTY)

  LIST     COPYPOOL
             (cpname)
             FASTREPLICATIONVOLS | NOVOLS | DUMPVOLS |
             ALLOVLS(
               GENERATION(gennum) | ALLVERS | TOKEN(token))
             SELECT(
```

```
            FASTREPLICATIONSTATE(RECOVERABLE | DUMPONLY |
                                 FAILED | NONE) |
            DUMPSTATE(ALLCOMPLETE | REQUIREDCOMPLETE |
                      PARTIAL | NONE))
        COPYPOOLBACKUPSTORAGEGROUP(cpbsgname)

LIST    DATASETNAME(dsname) | LEVEL(qualifier)
          BACKUPCONTROLDATASET | BOTH |
          MIGRATIONCONTROLDATASET
            SUMMARY
          INCLUDEPRIMARY
          OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
          SELECT(
            ACTIVE |
            AGE(mindays maxdays) |
            EMPTY |
            MIGRATIONLEVEL1 |
            MIGRATIONLEVEL2 |
            RETAINDAYS |
            VOLUME(volser)
          SMALLDATASETPACKING | NOSMALLDATASETPACKING
          VSAM)
LIST    DUMPCLASS(class)
             BACKUPCONTROLDATASET
             OUTDATASET(dsname) | SYSOUT(class) | TERMINAL

LIST    DUMPVOLUME(volser)
             BACKUPCONTROLDATASET
             DUMPCONTENTS(volser)
             SELECT(AVAILABLE UNAVAILABLE EXPIRED UNEXPIRED
             LIB NOLIB NORETENTIONLIMIT DUMPCLASS(class))
             OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
LIST    HOST(hostid)
             RESET
             OUTDATASET(dsname) | SYSOUT(class) | TERMINAL

LIST    MIGRATIONVOLUME |
        MIGRATIONLEVEL1 SELECT(OVERFLOW | NOOVERFLOW) |
        MIGRATIONLEVEL2(DASD | TAPE) | VOLUME(volser)
             BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET | BOTH
             ALLDUMPS
             OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
             SELECT(EMPTY)

LIST    PRIMARYVOLUME(volser)
             ALLDUMPS | BACKUPCONTENTS(nn)
             BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET | BOTH
             OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
             SELECT(MULTIPLEVOLUME VSAM)

LIST   TAPETABLEOFCONTENTS
             OUTDATASETNAME(dsname) | SYSOUT(class) | TERMINAL
             BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET | BOTH
             SELECT(
              MIGRATIONLEVEL2 | BACKUP | BOTH
              NOALTERNATEVOLUME | ALTERNATEVOLUME | FAILEDRECYCLE |
              FAILEDCREATE | EXCESSIVEVOLUMES | RECALLTAKEAWAY |
              DISASTERALTERNATEVOLUMES
              EMPTY | FULL | NOTFULL | ASSOCIATED | NOTASSOCIATED
              ERRORALTERNATE |
              CONNECTED(volser) | NOTCONNECTED
              LIB(ALTERNATE) | NOLIB(ALTERNATE) )

LIST   TAPETABLEOFCONTENTS(volser)
            OUTDATASET(dsname) | SYSOUT(class) | TERMINAL
            BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET | BOTH
            NODATASETINFORMATION | DATASETINFORMATION
```

```
  LIST     USER(userid)
               OUTDATASET(dsname) | SYSOUT(class) | TERMINAL

))LOG      data

))MIGRATE  DATASETNAME(dsname) | MIGRATIONLEVEL1 | PRIMARY |
           VOLUME(volser1 MIGRATE(days)
           DELETEBYAGE(days) | DELETEIFBACKEDUP(days)| MIGRATE(days))
                CONVERT(volser2 unittype2)
                DAYS(days)
                MIGRATIONLEVEL1
                MIGRATIONLEVEL2
                TERMINAL
                UNIT(unittype)

))PATCH    address
           data | BITS(bits)
                OUTDATASET(dsname)
                VERIFY(address data | BITS(bits))
                VOLSER(volser)))QUERY         ABARS
                ACTIVE
                ARPOOL(agname)
                AUTOPROGRESS
                BACKUP(ALL | DAILY(day) | SPILL | UNASSIGNED)
                CDSVERSIONBACKUP
                COMMONQUEUE(RECALL)
                CONTROLDATASETS
                COPYPOOL(cpname)
                CSALIMITS
                DATASETNAME(dsname) | REQUEST(reqnum) | USER(userid)
                MIGRATIONLEVEL2
                POOL
                RETAIN
                SECURITY
                SETSYS
                SPACE(volser ...)
                STARTUP
                STATISTICS
                TRAPS
                VOLUMEPOOL
                WAITING

))RECALL   dsname
                DAOPTION(SAMETRK | RELTRK | RELBLK)
                DFDSSOPTION(RETRY | VOLCOUNT(N(nn) | ANY) |
                    RETRY VOLCOUNT(N(nn) | ANY))
                FORCENONSMS
                UNIT(unittype)
                VOLUME(volser)

))RECOVER  dsname
                DAOPTION(SAMETRK | RELTRK | RELBLK)
                DATE(yyyy/mm/dd) TIME(hhmmss) |
                GENERATION(gennum)
                VERSION(vernum)
                DFDSSOPTION(
                    RETRY |
                    VOLCOUNT(N(nn) | ANY) |
                    RETRY VOLCOUNT(N(nn) | ANY))
                FORCENONSMS
                FROMDUMP(
                    DUMPCLASS(class) |
                    DUMPVOLUME(volser)
                        SOURCEVOLUME(volser))
                FROMVOLUME(volser)
                NEWNAME(newdsname)
```

```
                REPLACE
                RSA(keylabel)
                TOVOLUME(volser)
                    UNIT(unittype)

))RECOVER  *
                TOVOLUME(volser)
                UNIT(unittype)
                     DATE(date)
                     TARGETVOLUME(volser)
                     FROMDUMP(
                         DUMPCLASS(class) |
                         DUMPVOLUME(volser) |
                         DUMPGENERATION(dgennum)
                         APPLYINCREMENTAL)
                     RSA(keylabel)
                     TERMINAL
))RECYCLE  ALL | BACKUP | DAILY(day) | ML2 | SPILL | VOLUME(volser)
           CHECKFIRST(Y | N)
           DISPLAY | EXECUTE | VERIFY
                OUTDATASET(dsname)
                TAPELIST( PULLSIZE(size) TOTAL(count) PREFIX(prefix) |
                                                      FULLDSNAME(dsn))
                FORCE
                PERCENTVALID(pct)
                LIMIT(netfreed)

))RELEASE       ABACKUP(agname)
                ALL
                ARECOVER AGGREGATE(agname)|DATASETNAME(controlfiledsn)
                AUDIT
                AUTOMIGRATION
                BACKUP(AUTO
                       DSCOMMAND(
                                 DASD |
                                 TAPE))
                COMMONQUEUE(
                  RECALL(
                         SELECTION | PLACEMENT))
                DUMP(AUTO | FASTREPLICATIONBACKUP)
                EXPIREBV
                FRBACKUP
                FRRECOV(
                 DATASET | TAPE)
                HARDCOPY
                LIST
                LOG
                MIGRATION(AUTO)
                RECALL(
                       DASD |
                       TAPE
                           (TSO))
                RECOVER(TAPEDATASET)
                RECYCLE
                REPORT
                TAPECOPY
                TAPEREPL

))REPORT  DAILY
          DELETE
          FROMDATE(date)
          OUTDATASET(dsname) | SYSOUT(class)
          NOFUNCTION | FUNCTION
          (BACKUP | DELETE | MIGRATION(FROMANY | FROMMIGRATIONLEVEL1 |
                                       FROMPRIMARY)
                                      (TOANY | TOMIGRATIONLEVEL1 |
                                       TOMIGRATIONLEVEL2) |
```

```
        RECALL(FROMANY | FROMMIGRATIONLEVEL1 | FROMMIGRATIONLEVEL2)|
        RECOVER | RECYCLE(BACKUP | MIGRATION | ALL) | SPILL))
        SUMMARY
        TODATE(date)
        VOLUMES(volser ...)

))SETMIG   DATASETNAME(dsname) | LEVEL(qualifier) | VOLUME(volser)
             COMMANDMIGRATION | MIGRATION | NOMIGRATION

))SETSYS     ABARSPROCNAME(abarsprocname)
             ABARSACTLOGTYPE(SYSOUT(class) | DASD)
             ABARSACTLOGMSGLVL(FULL | REDUCED)
             ABARSBUFFERS(n)
             ABARSDELETEACTIVITY(Y | N)
             ABARSKIP(PPRC | XRC | NOPPRC | NOXRC)
             ABARSOPTIMIZE(1|2|3|4)
             ABARSPROCNAME(name)
             ABARSTAPES(STACK | NOSTACK)
             ABARSUNITNAME(unittype)
             ABARSVOLCOUNT(NONE | ANY)
             ACCEPTPSCBUSERID | NOACCEPTPSCBUSERID
             ACTLOGMSGLVL(FULL | EXCEPTIONONLY | REDUCED)
             ACTLOGTYPE(SYSOUT(class) | DASD)
             ARECOVERPERCENTUTILIZED(percent)
             ARECOVERREPLACE | NOARECOVERREPLACE
             ARECOVERTGTGDS(SOURCE|ACTIVITY|DEFERRED|ROLLEDOFF)
             ARECOVERUNITNAME(unittype)
             ARECOVERML2UNIT(unittype)
             AUTOBACKUPSTART(hhmm1(hhmm2(hhmm3)))
             AUTODUMPSTART(hhmm1(hhmm2(hhmm3)))
             AUTOMIGRATIONSTART(hhmm1(hhmm2(hhmm3)))
             BACKUP( ANY | DASD | TAPE(unittype)) | NOBACKUP
             BACKUPPREFIX(prefix)
             CDSVERSIONBACKUP
                  (BACKUPCOPIES(backupcopies)
                  DATAMOVER(HSM | DSS)
                  BACKUPDEVICECATEGORY(DASD |
                           TAPE
                           (PARALLEL | NOPARALLEL
                           DENSITY(density)
                           EXPIRATIONDATE(expirationdate) |
                           RETENTIONPERIOD(retentionperiod))
                           UNITNAME(unittype)))
                  BCDSBACKUPDSN(dsname)
                  JRNLBACKUPDSN(dsname)
                  MCDSBACKUPDSN(dsname)
                  OCDSBACKUPDSN(dsname))
             COMMONQUEUE(RECALL
                  (CONNECT(base_name) | DISCONNECT)
             COMPACT((ALL | NONE) | (DASDBACKUP | NODASDBACKUP)
                     (DASDMIGRATE | NODASDMIGRATE)
                     (TAPEBACKUP | NOTAPEBACKUP)
                     (TAPEMIGRATE | NOTAPEMIGRATE))

             COMPACTPERCENT(pct)

             CONVERSION((REBLOCKBASE | REBLOCKTOANY |
                        REBLOCKTOUNLIKE) | NOCONVERSION)

             CSALIMITS(ACTIVE(percent 1)
                       INACTIVE(percent 2)
                       MAXIMUM(Kbytes)
                       MWE(#mwes)) | NOCSALIMITS

             DAYS(days)
             DEBUG | NODEBUG
             DEFERMOUNT|NODEFERMOUNT
```

```
DENSITY(2 | 3 | 4)
DFHSMDATASETSERIALIZATION | USERDATASETSERIALIZATION
DISASTERMODE(Y|N)
DSBACKUP(DASDSELECTIONSIZE(maximum standard)
         DASD(TASKS(nn))
         TAPE(TASKS(nn)
              DEMOUNTDELAY(MINUTES(minutes)
                           MAXIDLETASKS(drives))))
DSSXMMODE(Y | N) |
  (BACKUP(Y | N)  CDSBACKUP(Y | N)  DUMP(Y | N)
   MIGRATION(Y | N)  RECOVERY(Y | N))
DUMPIO(1 | 2 | 3 | 4, 1 | 2 | 3 | 4)
DUPLEX(BACKUP(Y | N)
       MIGRATION(N | Y ERRORALTERNATE(CONTINUE | MARKFULL)))
EMERGENCY | NOEMERGENCY
ERASEONSCRATCH | NOERASEONSCRATCH
EXITOFF(modname, modname, ...)
EXITON(modname, modname, ...)
EXITS(abcdefghi)
EXPIREDDATASETS(SCRATCH | NOSCRATCH)
EXPORTESDS(CIMODE | RECORDMODE)
EXTENDEDTTOC(Y | N)
FASTREPLICATION(DATASETRECOVERY(PREFERRED |
                REQUIRED | NONE)
                FCRELATION(EXTENT | FULL)
                VOLUMEPAIRMESSAGES(YES | NO))
FREQUENCY(days)
INCREMENTALBACKUP(CHANGEDONLY | ORIGINAL)
INPUTTAPEALLOCATION(WAIT | NOWAIT)
INTERVALMIGRATION | NOINTERVALMIGRATION
JES2 | JES3
JOURNAL(RECOVERY | SPEED) | NOJOURNAL
MAXABARSADDRESSSPACE(number)
MAXBACKUPTASKS(tasks)
MAXCOPYPOOLTASKS(
   FRBACKUP(nn) FRRECOV(nn) DSS(nn))
MAXDSRECOVERTASKS(nn)
MAXDSTAPERECOVERTASKS(nn)
MAXDUMPRECOVERTASKS(nn)
MAXDUMPTASKS(nn)
MAXEXTENTS(extents)
MAXINTERVALTASKS(nn)
MAXMIGRATIONTASKS(nn)
MAXRECALLTASKS(nn)
MAXRECYCLETASKS(nn)
MAXSSMTASKS(CLEANUP(nn) TAPEMOVEMENT(mm))
MIGRATEPREFIX(prefix)
MIGRATIONCLEANUPDAYS(recalldays statdays reconnectdays)
MIGRATIONLEVEL1DAYS(days)
MIGUNITNAME(unittype)
ML1OVERFLOW(DATASETSIZE(dssize) THRESHOLD(threshold))
ML2PARTIALSNOTASSOCIATEDGOAL(nnn | NOLIMIT)
ML2RECYCLEPERCENT(pct)
MONITOR(BACKUPCONTROLDATASET(thresh)
        JOURNAL(thresh)
        MIGRATIONCONTROLDATASET(thresh)
        OFFLINECONTROLDATASET(thresh)
        SPACE | NOSPACE
        STARTUP | NOSTARTUP
        VOLUME | NOVOLUME)
MOUNTWAITTIME(minutes)
OBJECTNAMES(name1,name2,...)
ODMNOTIFICATIONLIMIT(limit)
ONDEMANDMIGRATION(Y | N)
OPTIMUMDASDBLOCKING | NOOPTIMUMDASDBLOCKING
OUTPUTTAPEALLOCATION(WAIT | NOWAIT)
PARTIALTAPE(MARKFULL | REUSE |
```

```
                 MIGRATION(MARKFULL | REUSE)
                 BACKUP(MARKFULL | REUSE))
             PDA(NONE | ON | OFF)
             PLEXNAME(plexname)
             PRIMARYSPMGMTSTART(hhmm1 (hhmm2))
             PROFILEBACKUP | NOPROFILEBACKUP
             PROMOTE(PRIMARYHOST(YES | NO) SSM(YES | NO))
             RACFIND | NORACFIND
             RECALL(ANYSTORAGEVOLUME(LIKE | UNLIKE) |
                   PRIVATEVOLUME(LIKE | UNLIKE))
             RECYCLEOUTPUT(BACKUP(unittype) MIGRATION(unittype))
             RECYCLEPERCENT(pct)
             RECYCLETAPEALLOCATION(WAIT | NOWAIT)
             REMOVECOMPACTNAMES(name1,name2,...)
             REQUEST | NOREQUEST
             SCRATCHFREQUENCY(days)
             SECONDARYSPMGMTSTART(hhmm1 (hhmm2))
             SELECTVOLUME(SCRATCH | SPECIFIC |
                MIGRATION(SCRATCH | SPECIFIC) |
                BACKUP(SCRATCH | SPECIFIC) |
                DUMP(SCRATCH | SPECIFIC) )
             SKIPABPRIMARY | NOSKIPABPRIMARY
             SMALLDATASETPACKING(tracks | KB(kilobytes)) |
             NOSMALLDATASETPACKING
             SMF(smfid) | NOSMF
             SOURCENAMES(name1,name2,...)
             SPILL(ANY | DASD | TAPE(unittype)) | NOSPILL
             SWAP | NOSWAP
             SYSOUT(class(copies forms))
             SYS1DUMP | NOSYS1DUMP
             TAPEDELETION(SCRATCHTAPE | HSMTAPE |
                 MIGRATION(SCRATCHTAPE | HSMTAPE)
                 BACKUP(SCRATCHTAPE | HSMTAPE)
                 DUMP(SCRATCHTAPE | HSMTAPE))
             TAPEFORMAT(SINGLEFILE)
             TAPEHARDWARECOMPACT | NOTAPEHARDWARECOMPACT
             TAPEINPUTPROMPT(MIGRATIONTAPES(YES | NO)
                             BACKUPTAPES(YES | NO)
                             DUMPTAPES(YES | NO))
             TAPEMAXRECALLTASKS(tasks)
             TAPEMIGRATION(DIRECT(TAPE(ANY | unittype)) |
                             ML2TAPE(TAPE(ANY | unittype)) |
                             NONE(ROUTETOTAPE(ANY | unittype))
                             RECONNECT(NONE |
                                       ALL |
                                       ML2DIRECTEDONLY))
             TAPEOUTPUTPROMPT(TAPECOPY(YES|NO))
             TAPESECURITY((EXPIRATION | EXPIRATIONINCLUDE)
                             PASSWORD (RACF | RACFINCLUDE))
             TAPESPANSIZE(nnn)
             TAPEUTILIZATION
               (UNITTYPE(unittype) PERCENTFULL(pct | NOLIMIT) |
               (LIBRARYBACKUP PERCENTFULL(pct | NOLIMIT) )    |
               (LIBRARYMIGRATION PERCENTFULL(pct | NOLIMIT) )
             UNITNAME(unittype)
             UNLOAD | NOUNLOAD
             USECYLINDERMANAGEDSPACE(Y | N)
             USERUNITTABLE(ES1,ES1OUT : ES2IN,ES3 : ES3) |
             NOUSERUNITTABLE
             VERSIONS(limit)
             VOLCOUNT(NONE | ANY)
             VOLUMEDUMP(NOCC | STANDARD |
                        CC | PREFERRED | REQUIRED |
                        VIRTUALPREFERRED | VIRTUALREQUIRED |
                        CACHEPREFERRED | CACHEREQUIRED)

))STOP       DUMP PROMOTE
```

```
))SWAPLOG

))TAPECOPY ALL | MIGRATIONLEVEL2 | BACKUP |
          ORIGINALVOLUMES(ovol1,ovol2,...ovoln) | INDATASET(dsname)
               ALTERNATEVOLUMES(avol1,avol2...avoln)
               EXPDT((cc)yyddd) | RETPD(nnnn)
               ALTERNATEUNITNAME(unittype1,unittype2) |
               ALTERNATE3590UNITNAME(unittype1,unittype2) |
               ALTERNATEUNITNAME(unittype1,unittype2)
                    ALTERNATE3590UNITNAME(unittype1,unittype2)

 ))TAPEREPL ALL | BACKUP |
           INDATASET(volrepl.list.dsname) |
           MIGRATION |
           ONLYDISASTERALTERNATES(
                                  RESET) |
           ORIGINALVOLUMES(ovol1,ovol2,...ovoln)
                ALTERNATEUNITNAME(unittype)
                ALTERNATEVOLUMES(avol1,avol2...avoln)
                DISASTERALTERNATEVOLUMES

))TRAP     ALL | module
               error code
               ABEND(ALWAYS | NEVER | ONCE) |
               LOG | OFF |
               SNAP(ALWAYS | NEVER | ONCE)

))UPDATEC  ALL | BACKUPCONTROLDATASET | MIGRATIONCONTROLDATASET |
          OFFLINECONTROLDATASET
               JOURNAL(dsname)
```

HSMLOG

HSMLOG helps you maintain and monitor the DFSMShsm environment. HSMLOG contains a job that prints the DFSMShsm log. Refer to Figure 31 for an example listing of the HSMLOG member.

```
//HSMLOG   JOB ?JOBPARM
//*
//*****************************************************************/
//*        THIS SAMPLE JOB PRINTS THE DFSMSHSM LOG               */
//*                                                              */
//* REPLACE THE ?UID VARIABLE IN THE FOLLOWING SAMPLE JOB WITH   */
//* THE NAME OF THE DFSMSHSM -AUTHORIZED USERID (1 TO 7 CHARS).  */
//*                                                              */
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS    */
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES         */
//* WILL BE USED FOR AUTHORIZATION CHECKING.)                    */
//*****************************************************************/
//*
//PRINTLOG EXEC PGM=ARCPRLOG
//ARCPRINT DD SYSOUT=*
//ARCLOG DD DSN=?UID.HSMLOGY1,DISP=OLD
//ARCEDIT DD DSN=?UID.EDITLOG,DISP=OLD
//*
//EMPTYLOG EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSUT2 DD DSN=?UID.HSMLOGY1,DISP=OLD
//SYSUT1 DD DUMMY,DCB=(?UID.HSMLOGY1)
/*
```

Figure 31. Example Listing of Member HSMLOG

Note: Do not compress the log data set used as input to the ARCPRLOG program. The log data set is created with RECFM=F but is opened by ARCPRLOG for update with RECFM=U, which is not allowed for compressed data sets.

HSMEDIT

HSMEDIT helps you maintain and monitor DFSMShsm. HSMEDIT contains a job that prints the edit log. Refer to Figure 32 and for an example of the HSMEDIT member and Figure 33 for a JCL example to send the edit log output to a data set.

```
//EDITLOG  JOB ?JOBPARM
//*
//***************************************************************/
//*           THIS JOB PRINTS THE EDIT-LOG DATA SET             */
//*                                                             */
//* REPLACE THE FOLLOWING ?UID VARIABLE WITH THE NAME OF THE     */
//* DFSMSHSM-AUTHORIZED USER (1 TO 7 CHARS).                    */
//*                                                             */
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS    */
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES WILL BE */
//* USED FOR AUTHORIZATION CHECKING.)                           */
//***************************************************************/
//*
//EDITLOG EXEC PGM=ARCPEDIT
//ARCPRINT DD SYSOUT=*
//ARCLOG DD DSN=?UID.EDITLOG,DISP=SHR
/*
```

Figure 32. Example Listing of Member HSMEDIT

To send the edit log output to a data set, change ARCPRINT to:

```
//ARCPRINT DD DSN=uid.EDITOUT,DISP=(NEW,CATLG),UNIT=unitname,
//     VOL=SER=volser,SPACE=spaceinfo,
//     DCB=(RECFM=FBA,LRECL=133,BLKSIZE=26600)
```

Figure 33. Example JCL to Send Output to a Data Set

ALLOCBK1

This sample job allocates four backup versions of each control data set. Ensure that backup version data sets are placed on volumes that are different from the volumes that the control data sets are on. Refer to Figure 34 on page 144 for an example listing of the ALLOCBK1 member.

Note: Four backup versions of each control data set are the default, but this number can be changed with the BACKUPCOPIES parameter of the SETSYS command.

The backup versions in this example are allocated on DASD volumes instead of tape volumes. Ensure that the following parameters are changed.

Parameter
Description

?BKUNIT1
Defines the unit type of the volume for the first control data set backup version.

?BKUNIT2
: Defines the unit type of the volume for the second control data set backup version.

?BKUNIT3
: Defines the unit type of the volume for the third control data set backup version.

?BKUNIT4
: Defines the unit type of the volume for the fourth control data set backup version.

?BKVOL1
: Defines the volume serial number of the volume for the first control data set backup version.

?BKVOL2
: Defines the volume serial number of the volume for the second control data set backup version.

?BKVOL3
: Defines the volume serial number of the volume for the third control data set backup version.

?BKVOL4
: Defines the volume serial number of the volume for the fourth control data set backup version.

?SCBVOL1
: Defines the storage class name for the backup versions.

?MCDFHSM
: Defines the management class name for the DFSMShsm data sets.

?MCDFHSM
: Defines the management class name for the DFSMShsm data sets.

?CDSSIZE
: Defines the number of cylinders allocated to any control data set backup version.

  Guideline: Initially, allocate 10 cylinders to run the starter set.

?JNLSIZE
: Defines the number of cylinders allocated to the journal dat sets.

?UID
: Defines the authorized user ID for the DFSMShsm-started procedure. An authorized user ID (?UID) must be from 1 to 7 characters long. This ID is also used as the high-level qualifier for the DFSMShsm managed-data sets.

During the edit, search for the character string starting with “REMOVE THE NEXT ...” and determine if the following JCL statements apply to your environment. If these JCL statements do not apply, ensure that they are removed from the data set.

```
//ALLOCBK1 JOB ?JOBPARM
//ALLOCBK EXEC PGM=IEFBR14
//*
//*****************************************************************/
//*    THIS SAMPLE JOB ALLOCATES AND CATALOGS THE CONTROL DATA SET*/
//*    BACKUP VERSION DATA SETS ON DASD VOLUMES.                  */
//*                                                               */
//*    ENSURE THAT BACKUP VERSION DATA SETS ARE PLACED ON VOLUMES */
//*    THAT ARE DIFFERENT FROM THE VOLUMES THAT THE CONTROL DATA  */
//*    SETS ARE ON.                                               */
//*                                                               */
//*    THIS SAMPLE JOB ALLOCATES FOUR BACKUP COPIES (THE DEFAULT) */
//*    FOR EACH CONTROL DATA SET.  IF YOU SPECIFY A DIFFERENT     */
//*    NUMBER OF BACKUP VERSIONS, ENSURE THAT YOU ALLOCATE A      */
//*    BACKUP COPY FOR EACH OF THE BACKUP VERSIONS YOU SPECIFY.   */
//*****************************************************************/
//*                                                               */
//*    EDIT THIS JCL TO REPLACE THE PARAMETERS DESCRIBED BELOW.   */
//*                                                               */
//*****************************************************************/
//*  PARAMETER PARAMETER DEFINITION
//*
//*  ?BKUNIT1   -  UNIT TYPE OF VOLUME TO CONTAIN THE FIRST CDS
//*                BACKUP VERSION.
//*  ?BKUNIT2   -  UNIT TYPE OF VOLUME TO CONTAIN THE SECOND CDS
//*                BACKUP VERSION.
//*  ?BKUNIT3   -  UNIT TYPE OF VOLUME TO CONTAIN THE THIRD CDS
//*                BACKUP VERSION.
//*  ?BKUNIT4   -  UNIT TYPE OF VOLUME TO CONTAIN THE FOURTH CDS
//*                BACKUP VERSION.
//*  ?BKVOL1    -  VOLUME SERIAL OF VOLUME TO CONTAIN THE FIRST CDS
//*                BACKUP VERSION.
//*  ?BKVOL2    -  VOLUME SERIAL OF VOLUME TO CONTAIN THE SECOND CDS
//*                BACKUP VERSION.
//*  ?BKVOL3    -  VOLUME SERIAL OF VOLUME TO CONTAIN THE THIRD CDS
//*                BACKUP VERSION.
//*  ?BKVOL4    -  VOLUME SERIAL OF VOLUME TO CONTAIN THE FOURTH CDS
//*                BACKUP VERSION.
//*  ?SCBVOL1   -  STORAGE CLASS NAME FOR BACKUP VERSIONS
//*  ?MCDFHSM   -  MANAGEMENT CLASS NAME OF THE HSM CONSTRUCT
//*
//*  ?CDSSIZE   -  NUMBER OF CYLINDERS ALLOCATED TO CDS BACKUP
//*                VERSIONS.
//*  ?JNLSIZE   -  NUMBER OF CYLINDERS ALLOCATED TO JOURNAL DATA
//*                SETS.
//*  ?UID       -  AUTHORIZED USER ID (1 TO 7 CHARS) FOR THE HSM-
//*                STARTED PROCEDURE. THIS WILL BE USED AS THE
//*                HIGH-LEVEL QUALIFIER OF HSM DATA SETS.
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS    */
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES WILL BE */
//* USED FOR AUTHORIZATION CHECKING.)                            */
//*****************************************************************/
//*
//******************************************************************/
//* THIS PROCEDURE ASSUMES A SINGLE CLUSTER MCDS.  IF MORE THAN    */
//* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER     */
//* CDS.                                                           */
//******************************************************************/
//*
//MCDSV1 DD DSN=?UID.MCDS.BACKUP.V0000001,DISP=(,CATLG),UNIT=?BKUNIT1,
//   VOL=SER=?BKVOL1,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//MCDSV2 DD DSN=?UID.MCDS.BACKUP.V0000002,DISP=(,CATLG),UNIT=?BKUNIT2,
//   VOL=SER=?BKVOL2,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//MCDSV3 DD DSN=?UID.MCDS.BACKUP.V0000003,DISP=(,CATLG),UNIT=?BKUNIT3,
//   VOL=SER=?BKVOL3,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//MCDSV4 DD DSN=?UID.MCDS.BACKUP.V0000004,DISP=(,CATLG),UNIT=?BKUNIT4,
//   VOL=SER=?BKVOL4,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//*
```

Figure 34. Example Listing of Member ALLOCBK1 Part 1 of 2

```
//*******************************************************************/
//* REMOVE THE NEXT FOUR DD STATEMENTS IF YOU DO NOT INTEND TO USE */
//* BACKUP AND DUMP                                                 */
//*                                                                 */
//* THIS PROCEDURE ASSUMES A SINGLE CLUSTER BCDS.  IF MORE THAN    */
//* ONE VOLUME IS DESIRED, FOLLOW THE RULES FOR A MULTICLUSTER     */
//* CDS.                                                            */
//*******************************************************************/
//*
//BCDSV1 DD DSN=?UID.BCDS.BACKUP.V0000001,DISP=(,CATLG),UNIT=?BKUNIT1,
//   VOL=SER=?BKVOL1,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//BCDSV2 DD DSN=?UID.BCDS.BACKUP.V0000002,DISP=(,CATLG),UNIT=?BKUNIT2,
//   VOL=SER=?BKVOL2,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//BCDSV3 DD DSN=?UID.BCDS.BACKUP.V0000003,DISP=(,CATLG),UNIT=?BKUNIT3,
//   VOL=SER=?BKVOL3,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//BCDSV4 DD DSN=?UID.BCDS.BACKUP.V0000004,DISP=(,CATLG),UNIT=?BKUNIT4,
//   VOL=SER=?BKVOL4,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//*
//*******************************************************************/
//* REMOVE THE NEXT FOUR DD STATEMENTS IF YOU DO NOT INTEND TO USE */
//* TAPE VOLUMES FOR DAILY BACKUP VOLUMES, SPILL BACKUP VOLUMES,   */
//* OR MIGRATION LEVEL 2 VOLUMES.                                   */
//*                                                                 */
//* THE OCDS MAY NOT EXCEED 1 VOLUME.                               */
//*******************************************************************/
//*
//OCDSV1 DD DSN=?UID.OCDS.BACKUP.V0000001,DISP=(,CATLG),UNIT=?BKUNIT1,
//   VOL=SER=?BKVOL1,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//OCDSV2 DD DSN=?UID.OCDS.BACKUP.V0000002,DISP=(,CATLG),UNIT=?BKUNIT2,
//   VOL=SER=?BKVOL2,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//OCDSV3 DD DSN=?UID.OCDS.BACKUP.V0000003,DISP=(,CATLG),UNIT=?BKUNIT3,
//   VOL=SER=?BKVOL3,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//OCDSV4 DD DSN=?UID.OCDS.BACKUP.V0000004,DISP=(,CATLG),UNIT=?BKUNIT4,
//   VOL=SER=?BKVOL4,SPACE=(CYL,(?CDSSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//*
//JRNLV1 DD DSN=?UID.JRNL.BACKUP.V0000001,DISP=(,CATLG),UNIT=?BKUNIT1,
//   VOL=SER=?BKVOL1,SPACE=(CYL,(?JNLSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//JRNLV2 DD DSN=?UID.JRNL.BACKUP.V0000002,DISP=(,CATLG),UNIT=?BKUNIT2,
//   VOL=SER=?BKVOL2,SPACE=(CYL,(?JNLSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//JRNLV3 DD DSN=?UID.JRNL.BACKUP.V0000003,DISP=(,CATLG),UNIT=?BKUNIT3,
//   VOL=SER=?BKVOL3,SPACE=(CYL,(?JNLSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
//JRNLV4 DD DSN=?UID.JRNL.BACKUP.V0000004,DISP=(,CATLG),UNIT=?BKUNIT4,
//   VOL=SER=?BKVOL4,SPACE=(CYL,(?JNLSIZE,5)),STORCLAS=?SCBVOL1,
//   MGMTCLAS=?MCDFHSM
```

Figure 35. Example Listing of Member ALLOCBK1 Part 2 of 2

ALLOSDSP

This sample job allocates a small-data-set-packing (SDSP) data set. Ensure that the following parameters are changed. Refer to Figure 36 on page 146 for an example listing of the ALLOSDSP member.

Parameter
Description

?SDSPCIS
Defines the control interval size value for the data component of the SDSP data set.

?SDSPUNT
Defines the unit type of the migration level 1 volume for the SDSP data set.

?SDSPVOL
Defines the volume serial number of the migration level 1 volume for the SDSP data set.

?UCATNAM
Defines the name and password of the user catalog for the DFSMShsm data sets.

?UID
Defines the authorized user ID for the DFSMShsm-started procedure. An authorized user ID must be from 1 to 7 characters long. This ID is also used as the high-level qualifier for the DFSMShsm-managed data sets.

```
//ALLOSDSP JOB ?JOBPARM
//*
//*******************************************************************/
//*   THIS SAMPLE JOB DEFINES AND INITIALIZES A SMALL-DATA-SET-   */
//*   PACKING DATA SET ON A MIGRATION LEVEL 1 VOLUME.             */
//*                                                               */
//*   THE DATA SET NAME IS REQUIRED TO BE ?UID.SMALLDS.V?SDSPVOL  */
//*   WHERE ?UID IS THE AUTHORIZED DFSMSHSM USER ID AND WHERE     */
//*   ?SDSPVOL IS THE VOLUME SERIAL NUMBER OF A MIGRATION LEVEL 1 */
//*   VOLUME.                                                     */
//*                                                               */
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS    *//
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES WILL BE *//
//* USED FOR AUTHORIZATION CHECKING.)                            *//
//*                                                               */
//*                                                               */
//*   AFTER YOU ALLOCATE THE SMALL-DATA-SET-PACKING DATA SET ON A */
//*   MIGRATION LEVEL 1 VOLUME, YOU MUST SPECIFY THE              */
//*   SMALLDATASETPACKING PARAMETER ON THE SETSYS COMMAND IN THE  */
//*   ARCCMD__ PARMLIB MEMBER AND ON THE ADDVOL COMMAND FOR THE   */
//*   VOLUME THAT CONTAINS THE SMALL-DATA-SET-PACKING DATA SET.   */
//*                                                               */
//*   CHANGE THE PARAMETERS LISTED BELOW TO VALID VALUES FOR YOUR */
//*   SYSTEM.                                                     */
//*******************************************************************/
//*  PARAMETER  -  PARAMETER DESCRIPTION
//*
//*  ?SDSPCIS   -  CONTROL INTERVAL SIZE VALUE FOR THE DATA COMPONENT
//*                OF THE SDSP DATA SET. IF THE SDSP UNIT TYPE IS
//*                3350, REPLACE ?SDSPCIS WITH 16384.  IF THE SDSP
//*                UNIT TYPE IS 3380, REPLACE WITH 20480.
//*                IF THE SDSP UNIT TYPE IS 3390, REPLACE WITH 26624.
//*  ?SDSPUNT   -  UNIT TYPE FOR MIGRATION LEVEL 1 VOLUME TO
//*                CONTAIN SMALL-DATA-SET-PACKING DATA SET.
//*  ?SDSPVOL   -  VOLUME SERIAL OF THE MIGRATION LEVEL 1 VOLUME
//*                TO CONTAIN SMALL-DATA-SET-PACKING DATA SET.
//*  ?UID       -  AUTHORIZED USER ID (1 TO 7 CHARS) FOR THE DFSMShsm
//*                START PROCEDURE, IN A NON FACILITY CLASS ENVIRONMENT.
//*                USED AS THE HIGH-LEVEL QUALIFIER OF DFSMSHSM
//*                DATA SETS.
//*
//* NOTE:         ENSURE THAT THE SMALL-DATA-SET-PACKING DATA SET
//*               IS NOT ALLOCATED ON AN SMS VOLUME. THE DATA SET
//*               SHOULD BE DEFINED IN A STORAGE CLASS FILTER TO
//*               EXCLUDE IT FROM AN SMS VOLUME AS THE OTHER DFSMShsm
//*               DATA SETS ARE.
//*******************************************************************/
```

Figure 36. Example Listing of Member ALLOSDSP Part 1 of 2

```
//*****************************************************************/
//* CREATE A ONE-RECORD, SEQUENTIAL DATA SET TO BE USED LATER TO  */
//* PRIME VSAM CLUSTERS DEFINED FOR DFSMSHSM.                     */
//*****************************************************************/
//*
//IEBDG    EXEC PGM=IEBDG
//SYSPRINT DD SYSOUT=*
//PRIMER   DD DSN=?UID.SDSP.PRIMER,DISP=(NEW,CATLG),UNIT=SYSDA,
// DCB=(RECFM=F,LRECL=200,BLKSIZE=200),SPACE=(TRK,(1))
//SYSIN    DD *
 DSD OUTPUT=(PRIMER)
 FD NAME=A,LENGTH=44,STARTLOC=1,PICTURE=44,                             X
   'Z999999999999999999999999999999999999999999'
 CREATE NAME=A
 END
//STEP1 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//INPRIMER DD DSN=?UID.SDSP.PRIMER,DISP=(OLD,DELETE)
//SDSP1 DD UNIT=?SDSPUNT,VOL=SER=?SDSPVOL,DISP=SHR
//SYSIN DD *
 DEFINE CLUSTER (NAME(?UID.SMALLDS.V?SDSPVOL) VOLUMES(?SDSPVOL) -
        CYLINDERS(5 0) FILE(SDSP1) -
        RECORDSIZE(2093 2093) FREESPACE(0 0) -
        INDEXED KEYS(45 0) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA -
        (CONTROLINTERVALSIZE(?SDSPCIS)) -
        INDEX -
        (CONTROLINTERVALSIZE(1024))
 REPRO ODS('?UID.SMALLDS.V?SDSPVOL') INFILE(INPRIMER)
```

Figure 37. Example Listing of Member ALLOSDSP Part 2 of 2

HSMPRESS

HSMPRESS helps you maintain and monitor DFSMShsm. HSMPRESS contains a job that reorganizes the control data sets. Refer to Figure 38 on page 149 for an example listing of the HSMPRESS member.

Ensure that the following parameters are changed before running HSMPRESS:

Parameter
: Description

?UID
: Defines the authorized user ID for the DFSMShsm-started procedure. An authorized user ID is up to 7 characters long. This ID is also used as the high-level qualifier for the DFSMShsm-managed data sets.

?NEW
: Defines an extention name for the new CDS being created from the import. Because of the name change made to the CDS, you must update the associated proclib member with the new CDS names.

If you wish to enlarge the CDS, preallocate a larger data set with the new size (either a new data set or delete the old data set and reallocate it with the same name), and then import. Again, if a new name is used, be sure to update the associated proclib member with the new CDS names.

This procedure assumes that the MCDS and BCDS are single cluster CDSs.

Guideline: Use the QUERY CONTROLDATASETS command to determine how full the control data sets are. Do not perform frequent “reorgs” of DFSMShsm control data sets. Unlike other databases, reorganizing DFSMShsm control data sets degrades performance for about three weeks. The only time that you should perform a reorganization is when you are moving or reallocating the data sets to a larger size or multiple clusters to account for growth. For these rare instances, use the HSMPRESS job.

Note: *x*CDSs DD statements with DISP=OLD keep other jobs from accessing the CDSs during the EXPORT/IMPORT process.
Attention: Before running this job, you must shut down all instances of DFSMShsm that share the CDSs.

```
//COMPRESS JOB ?JOBPARM
//*
//***************************************************************/
//*  THIS SAMPLE JOB IS TO COMPRESS THE CONTROL DATA SETS.      */
//*                                                             */
//*  NOTE:  BEFORE RUNNING THIS JOB, YOU MUST SHUT DOWN ALL     */
//*         ALL INSTANCES OF DFSMSHSM THAT SHARE THE CDS'S.     */
//*                                                             */
//* REPLACE THE ?UID VARIABLE WITH THE NAME OF THE DFSMSHSM-    */
//* AUTHORIZED USER ID (1 TO 7 CHARACTERS).                     */
//*                                                             */
//* (NOTE: UID AUTHORIZATION IS VALID IN A NON-FACILITY CLASS   */
//* ENVIRONMENT ONLY, OTHERWISE, FACILITY CLASS PROFILES WILL BE */
//* USED FOR AUTHORIZATION CHECKING.)                           */
//*                                                             */
//* REPLACE THE ?NEW VARIABLE WITH AN EXTENSION NAME FOR THE    */
//* NEW CDS BEING CREATED FROM THE IMPORT.  BECAUSE OF THE NAME  */
//* CHANGE MADE TO THE CDS, MAKE SURE TO UPDATE THE ASSOCIATED  */
//* PROCLIB MEMBER WITH THE NEW CDS NAME(S).                    */
//*                                                             */
//* IF YOU WISH TO ENLARGE THE CDS, PREALLOCATE A LARGER DATA   */
//* SET WITH THE NEW SIZE (EITHER A NEW DATA SET, OR DELETE THE  */
//* OLD DATA SET AND REALLOCATE WITH SAME NAME) THEN IMPORT.    */
//* AGAIN, IF A NEW NAME IS USED, BE SURE TO UPDATE THE         */
//* ASSOCIATED PROCLIB MEMBER WITH THE NEW CDS NAME(S).         */
//*                                                             */
//* THIS PROCEDURE ASSUMES THAT THE MCDS AND BCDS ARE SINGLE    */
//* CLUSTER CDS'S.                                              */
//*                                                             */
//* Note: xCDS'S DD STMTS WITH DISP=OLD WILL KEEP OTHER JOBS    */
//*  FROM ACCESSING THE CDS'S DURING THE EXPORT/IMPORT PROCESS.  */
//***************************************************************/
//*
//ALLOCATE EXEC PGM=IEFBR14
//EXPMCDS DD DSN=?UID.EXPORT.MCDS,DISP=(,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(20,2))
//EXPBCDS DD DSN=?UID.EXPORT.BCDS,DISP=(,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(20,2))
//EXPOCDS DD DSN=?UID.EXPORT.OCDS,DISP=(,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(20,2))
//*
//IDCAMS EXEC PGM=IDCAMS,REGION=512K
//MCDS DD DSN=?UID.MCDS,DISP=OLD
//BCDS DD DSN=?UID.BCDS,DISP=OLD
//OCDS DD DSN=?UID.OCDS,DISP=OLD
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
 LISTCAT ENT(?UID.MCDS ?UID.BCDS ?UID.OCDS) ALL
 EXAMINE NAME(?UID.MCDS) INDEXTEST
 IF LASTCC = 0 THEN -
        EXPORT ?UID.MCDS ODS(?UID.EXPORT.MCDS) TEMPORARY
 IF LASTCC = 0 THEN -
        IMPORT IDS(?UID.EXPORT.MCDS) ODS(?UID.MCDS.?NEW) -
        OBJECTS -
         ((?UID.MCDS -
         NEWNAME(?UID.MCDS.?NEW)) -
          (?UID.MCDS.DATA -
           NEWNAME(?UID.MCDS.?NEW.DATA)) -
          (?UID.MCDS.INDEX -
           NEWNAME(?UID.MCDS.?NEW.INDEX))) -
        CATALOG(?UCATNAM)
 IF MAXCC = 0 THEN -
        DELETE ?UID.EXPORT.MCDS NONVSAM
 EXAMINE NAME(?UID.BCDS) INDEXTEST
 IF LASTCC = 0 THEN -
        EXPORT ?UID.BCDS ODS(?UID.EXPORT.BCDS) TEMPORARY
```

Figure 38. Example Listing of Member HSMPRESS Part 1 of 2

```
IF LASTCC = 0 THEN -
        IMPORT IDS(?UID.EXPORT.BCDS) ODS(?UID.BCDS.?NEW) -
        OBJECTS -
         ((?UID.BCDS -
         NEWNAME(?UID.BCDS.?NEW)) -
          (?UID.BCDS.DATA -
           NEWNAME(?UID.BCDS.?NEW.DATA)) -
          (?UID.BCDS.INDEX -
           NEWNAME(?UID.BCDS.?NEW.INDEX))) -
        CATALOG(?UCATNAM)
 IF MAXCC = 0 THEN -
        DELETE ?UID.EXPORT.BCDS NONVSAM
 EXAMINE NAME(?UID.OCDS) INDEXTEST
 IF LASTCC = 0 THEN -
        EXPORT ?UID.OCDS ODS(?UID.EXPORT.OCDS) TEMPORARY
 IF LASTCC = 0 THEN -
        IMPORT IDS(?UID.EXPORT.OCDS) ODS(?UID.OCDS.?NEW) -
        OBJECTS -
         ((?UID.OCDS -
         NEWNAME(?UID.OCDS.?NEW)) -
          (?UID.OCDS.DATA -
           NEWNAME(?UID.OCDS.?NEW.DATA)) -
          (?UID.OCDS.INDEX -
           NEWNAME(?UID.OCDS.?NEW.INDEX))) -
        CATALOG(?UCATNAM)
 IF MAXCC = 0 THEN -
        DELETE ?UID.EXPORT.OCDS NONVSAM
 LISTCAT ENT(?UID.MCDS.NEW ?UID.BCDS.NEW ?UID.OCDS.NEW) ALL
/*
```

Figure 39. Example Listing of Member HSMPRESS Part 2 of 2

Chapter 7. DFSMShsm sample tools

For the customer who uses DFSMShsm, the SYS1.SAMPLIB sample tools member ARCTOOLS is shipped with the DFSMS licensed program.

ARCTOOLS job and sample tool members

The installation of DFSMShsm places a member called ARCTOOLS in the SYS1.SAMPLIB data set. Running the ARCTOOLS job creates the following partitioned data sets:

- HSM.SAMPLE.TOOL– Table 9 shows tasks that you might want to perform, the members of HSM.SAMPLE.TOOL that accomplish these tasks, as well as brief descriptions of each member.
- HSM.ABARUTIL.JCL — JCL used by ABARS utilities
- HSM.ABARUTIL.PROCLIB — JCL PROCs used by ABARS utilities
- HSM.ABARUTIL.DOCS — documentation for ABARS utilities

Table 9. Members of the HSM.SAMPLE.TOOL Data Set and Their Purposes

If you want to:	Use member:	That member is:
Extract data and generate a report from DCOLLECT records	DCOLREXX	A sample REXX exec that allows you to read records produced by DCOLLECT and create a simple report.
Determine key ranges when splitting large CDSs	SPLITER	A sample tool that allows you to determine appropriate key ranges for splitting the MCDS or BCDS. MCDS or BCDS record images are analyzed to determine the best 2-, 3-, or 4-way splits for the specified CDS. All records in the specified CDS are analyzed before the results are displayed.
	SPLITCDS	A sample JCL batch job that allows you to allocate the data sets necessary to analyze the MCDS and BCDS and to invoke SPLITER to perform the analysis. **Note:** For more information on SPLITCDS, see “Determining key ranges for a multicluster control data set” on page 35.
Convert data set masks to an assembler subprogram	GENMASK	A sample REXX exec that converts a list of data set masks to an assembler subprogram that can be included in data set related installation exits.
Issue the QUERY SETSYS command	QUERYSET	A sample REXX exec that issues QUERY SETSYS from an extended console with CART support. The results are returned in variables that can be processed.
Scan backup activity logs	SCANBLOG	A sample REXX exec that scans a week of backup activity logs and provides summary results.

Table 9. Members of the HSM.SAMPLE.TOOL Data Set and Their Purposes (continued)

If you want to:	Use member:	That member is:
Scan migration activity logs	SCANMLOG	A sample REXX exec that scans a week of migration activity logs and provides summary results.
Scan FSR data	SCANFSR	A sample REXX exec that scans FSR data in SMF and provides summary results.
Execute ABARS utilities	Other execs in HSM.SAMPLE.TOOL	Various REXX execs used by ABARS utilities.
Observe duplicate records when merging CDSs	PREMERGE	A sample JCL to assist in planning of a CDS merge.
Scan FSR data	FSRSTAT	A Sample REXX program that reads FSR data and presents statistical results.
Identify migrated VSAM keyrange data sets	FINDKRDS	A sample REXX exec that will read the MCDS and identify all migrated VSAM keyrange data sets.
Alter the management or storage class of a migrated data set	HALTER	A sample REXX program that modifies the STORCLAS or MGMTCLAS of a migrated data set.
Identify backed up VSAM key range data sets whose data mover was HSM.	BCDSKEYR	Sample JCL that reads the BCDS and identifies backed up VSAM key range data sets that used HSM as the data mover.

Chapter 8. Functional verification procedure

When you install the DFSMShsm program, the system modification program (SMP) installs ARCFVPST into the library SYS1.SAMPLIB. ARCFVPST is the functional verification procedure (FVP) for DFSMShsm.

Preparing to run the functional verification procedure

The FVP is a job stream that verifies the major functions of DFSMShsm. This FVP procedure contains seven separate jobs which are held with **TYPRUN=HOLD**. Release each separate job only after the preceding job has completed.

This FVP requires three DASD volumes: a DFSMShsm-managed volume, a migration level 1 volume, and a user volume. Use the user volume to exercise the backup and dump functions of DFSMShsm. The FVP also requires labeled tapes for backup, dump, and tape migration if you are verifying tape processing.

Steps for running the functional verification procedure

Before you begin: You must establish a DFSMShsm operating environment before running the Functional Verification Procedure. Refer to Chapter 6, “DFSMShsm starter set,” on page 101 for instructions on how to implement the starter set and establish a DFSMShsm operating environment.

Perform the following steps to run the FVP:

1. Run the job called CLEANUP. The JCL for this job step is located in the ARCFVPST member in SYS1.SAMPLIB.

   Example: Refer to “Cleanup job” on page 156 for an example of the JCL that is used in the CLEANUP job.
2. Define your automatic class selection filters. If you want DFSMShsm to process SMS-managed data sets during the FVP, you must define automatic class selection filters that allow allocation of those data sets (processed by the FVP) to SMS-managed storage.

   Example: You can use the high-level qualifier defined by the ?AUTHID parameter to determine if data sets are to be SMS managed. You can change the JCL to specify a storage class when it allocates data sets that are to be processed by DFSMShsm during the FVP. Refer to “Job step 1: Allocate a non-VSAM data set and a data set to prime VSAM data sets” on page 157 and “Job step 4: IDCAMS creates two VSAM data sets” on page 161 for an example of the data sets that are allocated by the FVP.
3. Edit the member ARCFVPST in SYS1.SAMPLIB. Place your appropriate job card parameters on the first job card.

   Example: Refer to Figure 40 on page 154 for an example of the JCL that is used by this job step.

```
//UPDJOB JOB ********** REPLACE WITH JOB CARD PARAMETERS ***************
//*
//********************************************************************/
//* PARTIAL LISTING OF THE DFSMSHSM FUNCTIONAL VERIFICATION PROCEDURE  */
//********************************************************************/
//*
//UPDSTEP EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=HSM.SAMPLE.CNTL(FVP),
//            DISP=OLD
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM='$$'
//*
```

Figure 40. Partial Listing of the DFSMShsm Functional Verification Procedure (FVP)

4. Submit the ARCFVPST JCL. This job step creates the member FVP in the partitioned data set named HSM.SAMPLE.CNTL.
5. Edit the member FVP in HSM.SAMPLE.CNTL. In member FVP are the JCL jobs and procedures that comprise the functional verification procedure. You can adapt this JCL for your system by making global changes to the parameters that begin with a question mark (?). Ensure that you have replaced all FVP parameters by doing a **FIND ?** to locate missed parameters. Refer to "FVP parameters" on page 155 for a description of the parameters that you need to modify before running the FVP JCL.

   Examples: Refer to the following JCL examples while you are editing the FVP member and preparing to run the FVP:
   - "Job step 1: Allocate a non-VSAM data set and a data set to prime VSAM data sets" on page 157
   - "Job step SDSPA: Create small-data-set-packing (SDSP) data set" on page 158
   - "Job step 2: Print the data sets created in STEP 1 of job ?AUTHIDA" on page 159
   - "Job step 3 (JOB B): Performing data set backup, migration, and recall" on page 160
   - "Job step 4: IDCAMS creates two VSAM data sets" on page 161
   - "Job step 5 (JOB C): Performing backup, migration, and recovery" on page 162
   - "Job steps 6, 7, and 8: Deleting and re-creating data sets" on page 163
   - "Job step 9 (JOB D): Recovering data sets" on page 164
   - "Job steps 10 and 11 (JOB E): Listing recovered data sets and recalling with JCL" on page 165
   - "Job step 12 (JOB F): Tape support" on page 166
   - "Job step 13 (JOB G): Dump function" on page 167
   -
6. Submit the FVP JCL for processing. If the FVP requires additional runs to verify the successful implementation of DFSMShsm, run the CLEANUP job before each rerun of the FVP.

   Example: If it is necessary to rerun the FVP, refer to "Cleanup job" on page 156 for an example of the JCL that is used in the CLEANUP job.
7. Run the job called FVPCLEAN. The JCL for this job step is located in the FVPCLEAN member in SYS1.SAMPLIB.

   Example: Refer to "FVPCLEAN job" on page 167 for an example of the JCL that is used by this job.
8. Start DFSMShsm.

FVP parameters

Member FVP contains the JCL jobs and job steps that comprise the FVP. You can edit the JCL to adapt it for your system by making global changes to the following parameters. You may need to make other changes as identified by the comments embedded in the JCL. Substitute values for your environment in the FVP parameters that begin with a question mark (?).

Restriction: If the FVP job is submitted while you are editing ARCFVPST, you must exit the edit so that the FVP job can update HSM.SAMPLE.CNTL with the new FVPCLEAN member.

?JOBPARM
Parameters to appear on the job card.

DFSMShsm verification steps issue authorized database commands. The user ID that appears in the job cards that follow must be given database authority by the control-authorized user. (See the starter set for your control-authorized user ID.)

?AUTHID
User ID of an authorized database user. This user ID will be used as the high-level qualifier of data sets on the DFSMShsm-managed volume. To run migration of a VSAM data set, this user ID must be an alias of an existing ICF catalog. See the discussion following ?UCATNAM for a description of migration of a VSAM data set.

This user ID is used to name the jobs in this procedure. To expedite the processing of this procedure, we suggest that this user ID be used to submit the jobs in this procedure from TSO.

?PASSWRD
Password of authorized data base user ?AUTHID.

?PRIVOL
Volume serial of a DFSMShsm-managed volume to be used as a primary volume.

?PRIUNT
Unit type of the primary DFSMShsm-managed volume.

?MIGVOL
Volume serial of a volume to be used as a level 1 migration volume.

?MIGUNT
Unit type of the level 1 migration volume.

Small-data-set-packing parameters

The FVP verifies the small-data-set-packing function (see Figure 43 on page 159). The following keywords apply only to the SDSP function verification:

?UID
Authorized user ID for the DFSMShsm-started procedure. This user ID is the same as the ID that you specified for ?UID in the starter set. It is the high-level qualifier for the small-data-set-packing data set allocated on the level 1 migration volume.

Guideline: Changing the ?UID parameter after your environment is set up can cause a problem when you are attempting to recall small-data-set-packing data sets. Because DFSMShsm uses the ?UID as the high-level qualifier for SDSPs, DFSMShsm knows migrated SDSPs only by their original ?UID.

?UCATNAM
: CATALOGNAME of the user catalog for DFSMShsm data sets (or ?UCATNAM CATALOGNAME/PASSWORD, if password is used). This must be the same value that you assigned to the ?UCATNAM parameter in the starter set. For more information, see "Starter set example" on page 109.

VSAM data set migration parameter

Migration of a VSAM data set requires an integrated catalog facility (ICF) catalog. An alias of ?AUTHID must be defined to an ICF user catalog. If the ICF catalog is not installed on the system, VSAM data set migration must be removed from the FVP. You can find the steps and commands that are associated with VSAM data set migration by searching for the data set name DATAV8 in the following jobs and job steps.

The following keyword applies only to "Job step 4: IDCAMS creates two VSAM data sets" on page 161:

?XCATNAM
: CATALOGNAME of the existing ICF user catalog with the alias of ?AUTHID (or ?XCATNAM CATALOGNAME/PASSWORD if password is used).

Tape support parameter

Tapes are required to verify tape migration and tape backup. If your system does not include those functions, remove the tape verification job (?AUTHIDF, see Figure 51 on page 166) from the FVP.

Mount requests for scratch tapes (PRIVAT) appear when DFSMShsm requires a tape for backup, migration, or dump processing. When a scratch tape is selected, it is automatically placed under DFSMShsm control. After the FVP has completed, issue the DELVOL PURGE command to remove scratch tapes from DFSMShsm control. The following parameter applies to tape verification and to dump verification:

?TAPEUNT
: Tape unit identification.

Dump function parameter

Tape and one DASD volume are required to verify DFSMSdss dump. If you do not intend to use the dump function, remove the ?AUTHIDG job from the FVP. The following parameter applies only to dump verification:

?DMPCLAS
: Name of the dump class to be defined for dump verification.

Jobs and job steps that comprise the functional verification procedure

This section provides JCL examples of the separate jobs and job steps that comprise the FVP. They are shown here to assist you while you are editing the JCL for your environment.

Cleanup job

The CLEANUP job (see Figure 41 on page 157) prepares your DFSMShsm environment for initially running the FVP. Run this job before you run the FVP. This job, if necessary, also prepares your DFSMShsm environment for a rerun of the FVP. If the FVP requires additional runs to verify the successful implementation of DFSMShsm, run this CLEANUP job before each rerun of the FVP.

If you have added tapes for backup, migration, or dump, issue the DELVOL PURGE command to delete those tapes.

Note: You might receive the message FIXCDS COMMAND FAILED with a return code of 0015 if the MCD is not present.

```
//?AUTHIDH JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//********************************************************************
//*                       CLEAN UP                                   *
//*                                                                  *
//* THIS JOB DELETES ALL DATA SETS THAT WERE CREATED AS A RESULT OF  *
//* A PREVIOUS PROCESSING OF THE FVP.                                *
//*                                                                  *
//* IF IT IS NECESSARY TO RESTART THE FVP, THIS CLEAN UP MUST BE RUN *
//* FIRST.                                                           *
//*                                                                  *
//* IF YOU HAVE ADDED TAPES FOR BACKUP, MIGRATION, OR DUMP, YOU      *
//* SHOULD DELVOL PURGE THOSE VOLUMES.                               *
//*                                                                  *
//* YOU SHOULD DELETE THE MCD RECORDS CREATED BY MIGRATION           *
//********************************************************************
//*
//STEP1 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
  DELETE '?AUTHID.PRIMER'
  DELETE '?AUTHID.DATA1'
  DELETE '?AUTHID.DATA2'
  DELETE '?AUTHID.DATA3'
  DELETE '?AUTHID.DATA4'
  DELETE '?AUTHID.DATA5'
  DELETE '?AUTHID.DATA6'
 HSEND WAIT DELVOL ?MIGVOL MIGRATION
 HSEND WAIT DELVOL ?PRIVOL PRIMARY
//*
//STEP2 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DELETE (?UID.SMALLDS.V?MIGVOL) CLUSTER PURGE
  DELETE (?AUTHID.DATAV7) CLUSTER PURGE
  DELETE (?AUTHID.DATAV8) CLUSTER PURGE
/*
//STEP3 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN   DD *
 /* You might receive the message FIXCDS COMMAND FAILED with a */
 /* return code of 0015 or message ARC0195I ERROR = RECORD     */
 /* NOT FOUND or both, if the MCD is not found.                */
  HSEND WAIT FIXCDS D ?AUTHID.DATA1 DELETE
  HSEND WAIT FIXCDS D ?AUTHID.DATA2 DELETE
  HSEND WAIT FIXCDS D ?AUTHID.DATA3 DELETE
  HSEND WAIT FIXCDS D ?AUTHID.DATA4 DELETE
  HSEND WAIT FIXCDS D ?AUTHID.DATAV8 DELETE
/*
$A
```

Figure 41. FVP Job That Cleans Up the Environment before the Initial Run of the FVP. Also run this job before any reruns of the FVP.

Job step 1: Allocate a non-VSAM data set and a data set to prime VSAM data sets

This FVP job step (Figure 42 on page 158) allocates non-VSAM data sets, and allocates a data set named **?AUTHID.PRIMER** that primes VSAM data sets created in "Job step 4: IDCAMS creates two VSAM data sets" on page 161.

```
//*******************************************************************
//* THIS STEP ALLOCATES NONVSAM DATA SETS USED FOR FUNCTION         *
//* VERIFICATION.  IT ALSO ALLOCATES "?AUTHID.PRIMER", A DATA SET   *
//* THAT IS USED TO PRIME THE VSAM DATA SETS CREATED BY STEP4.      *
//*******************************************************************
//*
//STEP1 EXEC PGM=IEBDG
//SYSPRINT DD SYSOUT=*
//DATA1 DD DSN=?AUTHID.DATA1,DISP=(,CATLG),UNIT=?PRIUNT,
// SPACE=(TRK,(1)),VOL=SER=?PRIVOL,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=400,DSORG=PS)
//DATA2 DD DSN=?AUTHID.DATA2,DISP=(,CATLG),UNIT=?PRIUNT,
// SPACE=(TRK,(1)),VOL=SER=?PRIVOL,
// DCB=(RECFM=VB,LRECL=100,BLKSIZE=500,DSORG=PS)
//DATA3 DD DSN=?AUTHID.DATA3(DATA),DISP=(,CATLG),UNIT=?PRIUNT,
//  VOL=SER=?PRIVOL,SPACE=(TRK,(2,,10)),
//  DCB=(RECFM=FB,LRECL=80,BLKSIZE=400,DSORG=PO)
//DATA4 DD DSN=?AUTHID.DATA4,DISP=(,CATLG),UNIT=?PRIUNT,
// SPACE=(TRK,(3)),VOL=SER=?PRIVOL,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=400,DSORG=PS)
//DATA5 DD DSN=?AUTHID.DATA5,DISP=(,CATLG),UNIT=?PRIUNT,
// SPACE=(CYL,(1)),VOL=SER=?PRIVOL,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
//PRIMER DD DSN=?AUTHID.PRIMER,DISP=(NEW,CATLG),UNIT=?PRIUNT,
// VOL=SER=?PRIVOL,DCB=(RECFM=F,LRECL=200,BLKSIZE=200),SPACE=(TRK,(1))
//SYSIN DD *
 DSD OUTPUT=(DATA1)
 FD  NAME=A,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=32,'FVP DATA1 JOB=FVP
   FVP1 STEP=STEP1'
 CREATE NAME=A
 END
 DSD OUTPUT=(DATA2)
 FD  NAME=B,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=32,'FVP DATA2 JOB=FVP
   FVP1 STEP=STEP1'
 CREATE NAME=B,QUANTITY=10
 END
 DSD OUTPUT=(DATA3)
 FD  NAME=C,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=32,'FVP DATA3 JOB=FVP
   FVP1 STEP=STEP1'
 CREATE NAME=C,QUANTITY=50
 END
 DSD OUTPUT=(DATA4)
 FD  NAME=D,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=32,'FVP DATA4 JOB=FVP
   FVP1 STEP=STEP1'
 CREATE NAME=D,QUANTITY=150
 END
 DSD OUTPUT=(DATA5)
 FD  NAME=E,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=32,'FVP DATA5 JOB=FVP
   FVP1 STEP=STEP1'
 CREATE NAME=E,QUANTITY=1000
 END
 DSD OUTPUT=(PRIMER)
 FD NAME=F,LENGTH=44,STARTLOC=1,PICTURE=44,                            X
   'Z9999999999999999999999999999999999999999999'
 CREATE NAME=F
 END
/*
```

Figure 42. FVP Step That Allocates Non-VSAM Data Sets and Data Set to Prime VSAM Data Sets

Job step SDSPA: Create small-data-set-packing (SDSP) data set

This FVP job step (Figure 43 on page 159) verifies that DFSMShsm can process small-data-set-packing data set ?UID.SMALLDS.V?MIGVOL. If you do not want to implement SDSP data sets, remove the allocation of ?UID.SMALLDS.V?MIGVOL and remove the SDSP keyword from the level 1 migration volume ADDVOL command and SETSYS command.

The data set name is required to be ?UID.SMALLDS.V?MIGVOL. Where:

?UID is the authorized DFSMShsm user ID (UID).

?MIGVOL is the volume serial number of the migration level 1 volume on which it resides.

Note: If the volume is on a 3350 device, change the CONTROLINTERVALSIZE value for the data component to 16384 (16K).

```
//*******************************************************************
//*          CREATE SMALL-DATA-SET-PACKING (SDSP) DATA SET           *
//*                                                                  *
//*   THIS JOB DEFINES AND INITIALIZES A SMALL-DATA-SET-PACKING DATA *
//*   SET ON A MIGRATION VOLUME.                                     *
//*                                                                  *
//*   THE DATA SET NAME IS REQUIRED TO BE "?UID.SMALLDS.V?MIGVOL",   *
//*   WHERE ?UID IS THE AUTHORIZED HSM USERID (UID) AND WHERE THE    *
//*   ?MIGVOL IS THE VOLUME SERIAL NUMBER OF THE MIGRATION LEVEL 1   *
//*   VOLUME ON WHICH IT RESIDES.                                    *
//*                                                                  *
//*   NOTE: IF THE VOLUME IS ON A 3350 DEVICE, CHANGE THE            *
//*   CONTROL INTERVAL SIZE VALUE FOR THE DATA COMPONENT TO          *
//*   16384 (16K) BYTES.                                             *
//*******************************************************************
//*
//SDSPA EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//SDSP1 DD UNIT=?MIGUNT,VOL=SER=?MIGVOL,DISP=SHR
//SYSIN DD *
 DEFINE CLUSTER (NAME(?UID.SMALLDS.V?MIGVOL) VOLUMES(?MIGVOL) -
        CYLINDERS(5 1) FILE(SDSP1) -
        RECORDSIZE(2093 2093) FREESPACE(0 0) -
        INDEXED KEYS(45 0) -
        UNIQUE NOWRITECHECK) -
        DATA -
        (CONTROLINTERVALSIZE(20480)) -
        INDEX -
        (CONTROLINTERVALSIZE(4096)) -
        CATALOG(?UCATNAM)
 REPRO IDS(?AUTHID.PRIMER) ODS(?UID.SMALLDS.V?MIGVOL)
```

Figure 43. FVP Procedure That Allocates SDSP Data Sets for the FVP

Job step 2: Print the data sets created in STEP 1 of job ?AUTHIDA

This FVP job step (Figure 44 on page 160) verifies that DFSMShsm can print the data sets created in “Job step 1: Allocate a non-VSAM data set and a data set to prime VSAM data sets” on page 157.

```
//**********************************************************************/
//*     THIS FVP JOB STEP PRINTS THE DATA SETS ALLOCATED IN STEP1      */
//**********************************************************************/
//*
//STEP2 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//DATA1 DD DSN=?AUTHID.DATA1,DISP=SHR
//DATA2 DD DSN=?AUTHID.DATA2,DISP=SHR
//DATA3 DD DSN=?AUTHID.DATA3(DATA),DISP=SHR
//DATA4 DD DSN=?AUTHID.DATA4,DISP=SHR
//DATA5 DD DSN=?AUTHID.DATA5,DISP=SHR
//SYSIN DD *
  PRINT INFILE(DATA1) COUNT(1)
  PRINT INFILE(DATA2) COUNT(1)
  PRINT INFILE(DATA3) COUNT(1)
  PRINT INFILE(DATA4) COUNT(1)
  PRINT INFILE(DATA5) COUNT(1)
//*
```

Figure 44. FVP Step That Prints the Data Sets Allocated in STEP1

Job step 3 (JOB B): Performing data set backup, migration, and recall

This FVP JOB and job step (Figure 45 on page 161) verifies that DFSMShsm can backup, migrate, and recall data sets. To add a primary volume in this format, you must specify an ADDVOL command in the PARMLIB member.

Example

```
ADDVOL ?PRIVOL UNIT(?PRIUNT) PRIMARY(AR)
```

Rules

1. You must start DFSMShsm before running this job.
2. When running in a JES3 environment, all ADDVOL commands must be placed in the ARCCMD*xx* PARMLIB member so that DFSMShsm recognizes them when it is started. If you are operating in a JES3 environment, ensure that you remove the ADDVOL commands from STEP3 (in the following sample job) and insert them in your DFSMShsm PARMLIB member.

Note:

1. You might receive the message FIXCDS COMMAND FAILED with a return code of 0015 if the MCD is not present.
2. The job in Figure 45 on page 161 assumes that migration is to a level-one (ML1) migration volume.

```
//?AUTHIDB JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//*********************************************************************/
//* THIS FVP JOB STEP VERIFIES DFSMSHSM BACKUP, MIGRATION, AND RECALL. */
//*********************************************************************/
//*
//STEP3 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 /* You might receive the message FIXCDS COMMAND FAILED with a */
 /* return code of 0015 or message ARC0195I ERROR = RECORD     */
 /* NOT FOUND or both, if the MCD is not found.                */
 HSEND WAIT SETSYS SDSP(1) FREQUENCY(0)
 HSEND WAIT ADDVOL ?PRIVOL UNIT(?PRIUNT) PRIMARY(AR)
 HSEND WAIT ADDVOL ?MIGVOL UNIT(?MIGUNT) MIG(ML1 SDSP)
 HBACKDS '?AUTHID.DATA1' WAIT
 HBACKDS '?AUTHID.DATA3' WAIT
 HBACKDS '?AUTHID.DATA5' WAIT
 HLIST LEVEL(?AUTHID) BOTH INCLUDEPRIM TERM
 HSEND WAIT FIXCDS D '?AUTHID.DATA1' PATCH(X'5D' X'88000F')
 HSEND WAIT FIXCDS D '?AUTHID.DATA2' PATCH(X'5D' X'88000F')
 HSEND WAIT FIXCDS D '?AUTHID.DATA3' PATCH(X'5D' X'88000F')
 HSEND WAIT FIXCDS D '?AUTHID.DATA4' PATCH(X'5D' X'88000F')
 HMIGRATE '?AUTHID.DATA1' WAIT
 HMIGRATE '?AUTHID.DATA2' WAIT
 HMIGRATE '?AUTHID.DATA3' WAIT
 HMIGRATE '?AUTHID.DATA4' WAIT
 HLIST LEVEL(?AUTHID) TERM
 HRECALL '?AUTHID.DATA1' WAIT
 HRECALL '?AUTHID.DATA3' WAIT
 HRECALL '?AUTHID.DATA4' WAIT
 HLIST LEVEL(?AUTHID) INCLUDEPRIM TERM
//*
```

Figure 45. FVP Job That Verifies DFSMShsm Backup, Migration, and Recall Processing

Job step 4: IDCAMS creates two VSAM data sets

This FVP job step (Figure 46 on page 162) creates two VSAM data sets (DATAV7 and DATAV8) for STEP5. VSAM migration requires that DATAV8 be associated with an integrated catalog facility (ICF) catalog. You can remove the definition of DATAV8 if you do not want to test VSAM data set migration.

```
//********************************************************************
//*            THIS STEP CREATES TWO VSAM DATA SETS.                *
//*                                                                 *
//*     NOTE  - MIGRATION OF A VSAM DATA SET REQUIRES AN INTEGRATED  *
//*            CATALOG FACILITY (ICF) CATALOG.                      *
//*            THE DATA SET WITH THE NAME DATAV8 IS USED BY A       *
//*            SUBSEQUENT STEP FOR MIGRATION.  YOU CAN REMOVE THE   *
//*            DEFINITION OF DATAV8 IF YOU ARE NOT GOING TO TEST    *
//*            VSAM DATA SET MIGRATION.                             *
//********************************************************************
//*
//STEP4   EXEC PGM=IDCAMS,REGION=512K
//DD1       DD DISP=OLD,UNIT=?PRIUNT,VOL=SER=?PRIVOL
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
//*
//*******************************************************************/
//* DEFINE A VSAM CLUSTER FOR USE IN BACKUP. THIS DOES NOT REQUIRE AN  */
//* INTEGRATED CATALOG FACILITY (ICF) CATALOG.                        */
//*******************************************************************/
//*
     DEFINE CLUSTER -
            (NAME(?AUTHID.DATAV7) -
            VOLUMES(?PRIVOL) -
            FILE(DD1) -
            UNIQUE -
            INDEXED -
            RECORDS(50 50) -
            KEYS(2 1) -
            RECORDSIZE(800 800))
     REPRO -
       INDATASET(?AUTHID.PRIMER) -
       OUTDATASET(?AUTHID.DATAV7)
     PRINT -
       INDATASET(?AUTHID.DATAV7)
//*
//*******************************************************************/
//* DEFINE A VSAM CLUSTER FOR USE IN MIGRATION.  THIS REQUIRES AN     */
//* INTEGRATED CATALOG FACILITY (ICF) CATALOG.                        */
//*******************************************************************/
//*
     DEFINE CLUSTER -
            (NAME(?AUTHID.DATAV8) -
            UNIQUE -
            NONINDEXED -
            FILE(DD1) -
            RECORDS(50 50) -
            VOLUMES(?PRIVOL)  -
            RECORDSIZE(800 800))
     REPRO -
       INDATASET(?AUTHID.PRIMER) -
       OUTDATASET(?AUTHID.DATAV8)
     PRINT -
       INDATASET(?AUTHID.DATAV8)
//*
```

Figure 46. FVP Step That Allocates Two VSAM Data Sets for Verification Testing

Job step 5 (JOB C): Performing backup, migration, and recovery

This FVP JOB and job step (Figure 47 on page 163) verifies the DFSMShsm backup, migration, and recovery functions.

This job step enables you to perform the following tasks:

- Back up a VSAM data set (DATAV7)
- Migrate a VSAM data set (DATAV8)

- List the contents of the MCDS and BCDS
- Recover a VSAM data set (DATAV7)
- Recall a VSAM data set (DATAV8)
- List the contents of the MCDS and BCDS

Rule: You must define the data set with the name DATAV8 in an ICF catalog. If your system does not have ICF catalog support, remove the commands referring to DATAV8.

Note: You might receive the message FIXCDS COMMAND FAILED with a return code of 0015 if the MCD is not present.

```
//?AUTHIDC JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//********************************************************************
//*     THIS STEP BACKS UP, RECOVERS, MIGRATES, AND RECALLS          *
//*     VSAM DATA SETS.                                              *
//*                                                                  *
//*     STEP5  - BACKDS A VSAM DATA SET (DATAV7)                     *
//*            - MIGRATE A VSAM DATA SET (DATAV8)                    *
//*            - LIST THE CONTENTS OF THE MCDS AND BCDS              *
//*            - RECOVER A VSAM DATA SET (DATAV7)                    *
//*            - RECALL A VSAM DATA SET (DATAV8)                     *
//*            - LIST THE CONTENTS OF THE MCDS AND BCDS              *
//*                                                                  *
//*    NOTE    - THIS STEP REQUIRES THE DATA SET WITH THE NAME       *
//*              DATAV8 TO BE DEFINED IN A DF/EF CATALOG. IF YOUR    *
//*              SYSTEM DOES NOT HAVE DF/EF CATALOG SUPPORT,         *
//*              REMOVE THE COMMANDS REFERRING TO DATAV8.            *
//********************************************************************
//*
//STEP5  EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 /* You might receive the message FIXCDS COMMAND FAILED with a */
 /* return code of 0015 or message ARC0195I ERROR = RECORD     */
 /* NOT FOUND or both, if the MCD is not found.                */
HBACKDS '?AUTHID.DATAV7' WAIT
HSEND WAIT FIXCDS D '?AUTHID.DATAV8' PATCH(X'5D' X'88000F')
HMIGRATE '?AUTHID.DATAV8' WAIT
HLIST LEVEL(?AUTHID) BOTH INCLUDEPRIM TERM
HRECOVER '?AUTHID.DATAV7' REPLACE WAIT
HRECALL '?AUTHID.DATAV8' WAIT
HLIST LEVEL(?AUTHID) BOTH INCLUDEPRIM TERM
//*
```

Figure 47. FVP Job That Verifies DFSMShsm Backup, Migration, Recall, and Recovery of VSAM Data Sets

Job steps 6, 7, and 8: Deleting and re-creating data sets

This FVP job step (Figure 48 on page 164) verifies that DFSMShsm can delete two data sets and then re-create them with recovered data.

```
//*
//*******************************************************************
//* THESE STEPS DELETE TWO OF THE DATA SETS AND RECREATE THEM WITH   *
//* DIFFERENT DATA SO THAT RECOVERED DATA CAN BE TESTED.            *
//*                                                                 *
//*     STEP6 - IEFBR14 DELETE "?AUTHID.DATA1" AND "?AUTHID.DATA5".  *
//*     STEP7 - IEBDG RECREATE "?AUTHID.DATA1" AND "?AUTHID.DATA5"   *
//*             WITH NEW DATA.                                      *
//*     STEP8 - AMS LIST "?AUTHID.DATA1" AND "?AUTHID.DATA5"         *
//*******************************************************************
//*
//STEP6 EXEC PGM=IEFBR14
//DD1 DD DSN=?AUTHID.DATA1,DISP=(OLD,DELETE)
//DD2 DD DSN=?AUTHID.DATA5,DISP=(OLD,DELETE)
//STEP7 EXEC PGM=IEBDG
//SYSPRINT DD SYSOUT=*
//DATA1 DD DSN=?AUTHID.DATA1,DISP=(,CATLG),
// UNIT=?PRIUNT,VOL=SER=?PRIVOL,
// SPACE=(TRK,(1)),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400,DSORG=PS)
//DATA5 DD DSN=?AUTHID.DATA5,DISP=(,CATLG),
// UNIT=?PRIUNT,VOL=SER=?PRIVOL,
// SPACE=(CYL,(1)),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
//SYSIN DD *
 DSD OUTPUT=(DATA1)
 FD  NAME=A,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=36,'FVP DATA1 NEW JOB
   =FVPFVP2 STEP=STEP7'
 CREATE NAME=A
 END
 DSD OUTPUT=(DATA5)
 FD  NAME=E,LENGTH=80,STARTLOC=1,FILL=' ',PICTURE=36,'FVP DATA5 NEW JOB
   =FVPFVP2 STEP=STEP7'
 CREATE NAME=E,QUANTITY=1000
 END
//STEP8 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//DATA1 DD DSN=?AUTHID.DATA1,DISP=SHR
//DATA5 DD DSN=?AUTHID.DATA5,DISP=SHR
//SYSIN DD *
 PRINT INFILE(DATA1) COUNT(1)
 PRINT INFILE(DATA5) COUNT(1)
//*
```

Figure 48. FVP Step That Verifies That DFSMShsm Can Delete and Recover Data Sets

Job step 9 (JOB D): Recovering data sets

This FVP JOB and job step (Figure 49 on page 165) verifies that DFSMShsm can recover data sets ?AUTHID.DATA1 and ?AUTHID.DATA5 from backup in the following sequence:

1. ?AUTHID.DATA1 is recovered and replaces an online copy.
2. ?AUTHID.DATA5 is recovered as a new data set, ?AUTHID.DATA6.

```
//?AUTHIDD JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//********************************************************************
//* THIS STEP RECOVERS "?AUTHID.DATA1" AND "?AUTHID.DATA5" FROM      *
//* HSM BACKUP.                                                      *
//*                                                                  *
//*          - RECOVER "?AUTHID.DATA1" AND REPLACE ONLINE COPY.      *
//*          - RECOVER "?AUTHID.DATA5" AS A NEW DATA SET             *
//*            "?AUTHID.DATA6".                                      *
//********************************************************************
//*
//STEP9 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 HRECOVER '?AUTHID.DATA1' REPLACE WAIT
 HRECOVER '?AUTHID.DATA5' NEWNAME('?AUTHID.DATA6') WAIT
//*
```

Figure 49. FVP Job That Verifies that DFSMShsm Can Recover Data Sets

Job steps 10 and 11 (JOB E): Listing recovered data sets and recalling with JCL

This FVP JOB and job step (Figure 50) verifies that DFSMShsm can list recovered data sets ?AUTHID.DATA1, ?AUTHID.DATA5, and ?AUTHID.DATA6 and recall migrated data set ?AUTHID.DATA2 when it is referred to in the JCL.

```
//?AUTHIDE JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//********************************************************************
//* THIS STEP LISTS "?AUTHID.DATA1", "?AUTHID.DATA5", AND            *
//* "?AUTHID.DATA6" AS RECOVERED AND FORCES "?AUTHID.DATA2" TO BE    *
//* RECALLED VIA JCL REFERENCE.                                      *
//*                                                                  *
//*    STEP10 - AMS LIST "?AUTHID.DATA1", "?AUTHID.DATA5", AND       *
//*             "?AUTHID.DATA6".                                     *
//*             "?AUTHID.DATA1" SHOULD HAVE OLD DATA SINCE REPLACE   *
//*             MADE "?AUTHID.DATA5".                                *
//*             "?AUTHID.DATA5" SHOULD HAVE NEW DATA SINCE NO        *
//*             REPLACE MODE.                                        *
//*             "?AUTHID.DATA6" SHOULD HAVE THE OLD VERSION OF       *
//*             "?AUTHID.DATA5"                                      *
//*                                                                  *
//*    STEP11 - IEFBR14 FORCE ALLOCATION TO RECALL "?AUTHID.DATA2".  *
//********************************************************************
//*
//STEP10 EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=*
//DATA1 DD DSN=?AUTHID.DATA1,DISP=SHR
//DATA5 DD DSN=?AUTHID.DATA5,DISP=SHR
//DATA6 DD DSN=?AUTHID.DATA6,DISP=SHR
//SYSIN DD *
 PRINT INFILE(DATA1) COUNT(1)
 PRINT INFILE(DATA5) COUNT(1)
 PRINT INFILE(DATA6) COUNT(1)
//STEP11 EXEC PGM=IEFBR14
//DD1 DD DSN=?AUTHID.DATA2,DISP=OLD
//*
```

Figure 50. FVP Job That Lists Recovered Data Sets and Verifies That a Data Set Is Recalled When It Is Referred To

Job step 12 (JOB F): Tape support

This FVP JOB and job step (Figure 51) verifies DFSMShsm tape support. If you do not use tape, remove this job from the FVP.

Note: You might receive the message FIXCDS COMMAND FAILED with a return code of 0015 if the MCD is not present.

This job step enables you to perform the following tasks:

- Back up ?AUTHID.DATA5 to a level 1 DASD volume
- Change to direct tape migration
- Migrate ?AUTHID.DATA5
- Change to DASD migration
- List migration volumes
- Recall ?AUTHID.DATA5
- Back up volume ?PRIVOL to tape
- List backup volumes
- Recover ?PRIVOL from tape

```
//?AUTHIDF JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//*********************************************************************
//* THIS JOB PERFORMS THE VERIFICATION OF TAPE SUPPORT.  IF TAPE    *
//* IS NOT USED, THIS JOB SHOULD BE REMOVED FROM THE PROCEDURE.     *
//*********************************************************************
//*
//*********************************************************************
//*   THE FOLLOWING STEP WILL:                                        *
//*           - BACKUP "?AUTHID.DATA5" TO ML1 DASD.                   *
//*           - CHANGE TO DIRECT TAPE MIGRATION.                      *
//*           - MIGRATE "?AUTHID.DATA5".                              *
//*           - CHANGE TO DASD MIGRATION.                             *
//*           - LIST MIGRATION VOLUMES.                               *
//*           - RECALLS "?AUTHID.DATA5".                              *
//*           - BACKS UP VOLUME ?PRIVOL TO TAPE.                      *
//*           - LIST BACKUP VOLUMES.                                  *
//*           - RECOVER ?PRIVOL FROM TAPE                             *
//*********************************************************************
//*
//STEP12 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 /* You might receive the message FIXCDS COMMAND FAILED with a */
 /* return code of 0015 or message ARC0195I ERROR = RECORD     */
 /* NOT FOUND or both, if the MCD is not found.                */
 HSEND WAIT SETSYS UNIT(?TAPEUNT)
 HBACKDS '?AUTHID.DATA5' WAIT
 HSEND WAIT SETSYS TAPEMIGRATION(DIRECT(TAPE(?TAPEUNT)))
 HSEND WAIT FIXCDS D '?AUTHID.DATA5' PATCH(X'5D' X'88000F')
 HMIGRATE '?AUTHID.DATA5' WAIT
 HSEND WAIT SETSYS TAPEMIGRATION(NONE)
 HSEND WAIT LIST MIGRATIONVOLUME
 HRECALL '?AUTHID.DATA5' WAIT
 HSEND WAIT BACKVOL VOLUME(?PRIVOL) UNIT(?PRIUNT) B(TAPE) TOTAL
 HSEND WAIT LIST BACKUPVOLUME
 HSEND WAIT RECOVER * TOVOLUME(?PRIVOL) UNIT(?PRIUNT)
//*
```

Figure 51. FVP Job That Verifies DFSMShsm Tape Processing Functions

Job step 13 (JOB G): Dump function

This FVP JOB and job step (Figure 52) verifies that DFSMShsm can dump a primary volume (?PRIVOL) to a dump class (?DMPCLAS). If you do not want dump processing, remove this job step from the FVP.

```
//?AUTHIDG JOB ?JOBPARM,
//    TYPRUN=HOLD
//*
//********************************************************************
//* THIS JOB VERIFIES THE DFDSS DUMP FUNCTION. IF THE DFDSS DUMP     *
//* FUNCTION IS NOT TO BE USED IN YOUR SYSTEM, REMOVE THIS JOB FROM  *
//* THE PROCEDURE.                                                   *
//*                                                                  *
//* THE DUMP FUNCTION USES THE DUMP CLASS NAMED "?DMPCLAS". THE      *
//* VOLUME ?PRIVOL IS DUMPED TO THAT CLASS.                          *
//********************************************************************
//*
//STEP13 EXEC PGM=IKJEFT01,REGION=512K
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 HSEND WAIT DEFINE DUMPCLASS(?DMPCLAS FREQUENCY(90) RETPD(356) -
      NOAUTOREUSE NODATASETRESTORE NORESET -
      DISPOSITION('FVP-END') VTOCCOPIES(0))
 HSEND WAIT BACKVOL VOL(?PRIVOL) DUMP(DUMPCLASS(?DMPCLAS) -
      RETPD(NOLIMIT)) UNIT(?PRIUNT)
 HSEND WAIT LIST DUMPCLASS(?DMPCLAS)
 HSEND WAIT RECOVER * FROMDUMP(DUMPCLASS(?DMPCLAS)) -
      TOVOL(?PRIVOL) U(?PRIUNT)
 HSEND WAIT LIST DUMPCLASS(?DMPCLAS)
$$
```

Figure 52. FVP Job That Verifies DFSMShsm Dump Function Processing

FVPCLEAN job

This job creates a new member, FVPCLEAN, in HSM.SAMPLE.CNTL (Figure 53). Run the FVPCLEAN job after a successful run of the FVP to remove data sets allocated by the FVP and to remove all DFSMShsm-owned DASD volumes that are added by the FVP. Remove tapes from DFSMShsm's control with the DELVOL PURGE command.

```
//********************************************************************
//* THIS STEP CREATES THE MEMBER FVPCLEAN IN THE HSM.SAMPLE.CNTL     *
//* DATA SET.                                                        *
//********************************************************************
//*
//UPDSTEP EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD UNIT=SYSDA,
//            DSN=HSM.SAMPLE.CNTL(FVPCLEAN),
//            DISP=OLD
//SYSIN    DD DUMMY
//SYSUT1   DD DATA,DLM='$A'
//*
```

Figure 53. Sample JCL That Creates the Job That Cleans Up the Environment after a Successful Run of the FVP

Chapter 9. Authorizing and protecting DFSMShsm commands and resources

The DFSMShsm program manages data on DASD and tapes. The security of data on DFSMShsm-managed DASD and tapes and the security of the DFSMShsm environment itself are important considerations. When you implement DFSMShsm, you must determine how to protect your data and you must determine who will have authority to access that data.

Resource Access Control Facility (RACF) is used to protect resources and authorize users. RACF's objective is to protect system and user resources. **Example:** These resources includes terminal lines, tapes, data, programs, TSO (Time Sharing Option) procedures, TSO logon procedures, and logon access.

RACF protection is achieved by creating profiles. The profiles are assigned to users, groups of users, and resources. The profiles enable RACF to determine if the user or group has authority to access a given resource.

The descriptions that follow discuss the steps to implement a secure DFSMShsm environment (**Example:** Protection for DFSMShsm resources, control data sets, logs, small-data-set-packing (SDSP) data sets, migration data, backup data, and DFSMShsm-managed tapes).

The following information is divided into the following tasks:
- "Identifying DFSMShsm to RACF"
- "Identifying DFSMShsm to z/OS UNIX System Services" on page 173
- "Authorizing and protecting DFSMShsm commands in the RACF FACILITY class environment" on page 173
- "Authorizing commands issued by an operator" on page 180
- "Protecting DFSMShsm resources" on page 180
- "Authorizing users to access DFSMShsm resources" on page 184
- "Authorizing and protecting DFSMShsm resources in a nonsecurity environment" on page 186
- "Protecting DFSMShsm commands in a nonsecurity environment" on page 186

For additional information about RACF, see *z/OS DFSMShsm Storage Administration*.

Identifying DFSMShsm to RACF

To apply RACF protection to DFSMShsm processing, you must add DFSMShsm (a started procedure) to the RACF started-procedures table. You do this by creating a user ID for the DFSMShsm startup procedure and entering that user ID and the DFSMShsm startup procedure name into the RACF started-procedures table.

The DFSMShsm startup procedure in the member DFSMShsm in SYS1.PROCLIB provides the start procedure for the DFSMShsm primary address space. This procedure has system-generated JOB statements that do not contain the USER, GROUP, or PASSWORD parameters.

Before you begin: In a RACF environment, you must do the following before starting the DFSMShsm address space:
1. Create a RACF user ID for DFSMShsm.
2. Update the RACF started-procedure table with the name and ID of the DFSMShsm startup procedure.

If you are using aggregate backup and recovery support (ABARS), you must also:
1. Create a RACF user ID for ABARS.
2. Update the RACF started-procedure table with the name and ID of the ABARS startup procedure.

Creating user IDs

RACF requires a user ID for DFSMShsm and a user ID for aggregate backup and recovery support (ABARS).

Specifying a RACF user ID for DFSMShsm

If DFSMShsm and RACF are installed on the same processing unit, you must create a RACF user ID for DFSMShsm. RACF associates the DFSMShsm user ID with a RACF profile for DFSMShsm. The profile allows DFSMShsm to bypass RACF protection during migration and backup of user data sets. If a product functionally equivalent to RACF is being used, consult that product's publication for implementation of that product.

You should use the ID you have specified as the *UID* in DFSMShsm's startup procedure as the user ID on the ADDUSER command.

If you specify . . .	Then . . .
ADDUSER userid DFLTGRP (groupname)	a RACF user ID is created for the person identified with *userid*. **Note:** If you do not specify the default group on the RACF ADDUSER command, the current connect group of the user issuing the ADDUSER command is used as the default group for the user ID. The RACF user ID should not have the automatic data set protection (ADSP) attribute.

Note: If you are using remote sharing functions (RRSF) to propagate RACF commands to remote hosts, it is suggested that you define the RACF user ID for DFSMShsm with the SPECIAL and OPERATIONS attributes on all of the recipient systems.

Specifying a RACF user ID for ABARS

You can create a separate RACF user ID for ABARS processing, or you can use the same user ID that you used for DFSMShsm (see Table 10 on page 172). In the starter set topic, "Starter set example" on page 109, the name DFHSMABR has been chosen for the ABARS ID.

Associating a user ID with a started task

There are two methods to associate a user ID with a started task: 1) the RACF started procedures table (ICHRIN03) and 2) the RACF STARTED class. The RACF started procedures table method is described here first. However, you must use one of the two methods.

Method 1–RACF started procedures table (ICHRIN03)

Because DFSMShsm runs as a started task, you can modify the RACF started-procedures table (ICHRIN03) before starting the DFSMShsm address space.

Include in that table the name of each procedure used to start the DFSMShsm primary address space and the name of each procedure used to start an aggregate backup and recovery secondary address space. Associate the name of each procedure with the RACF user ID you have defined for DFSMShsm or ABARS. For more information about coding and replacing the RACF started-procedures module, refer to *z/OS Security Server RACF System Programmer's Guide* and *z/OS DFSMShsm Storage Administration*. After you have replaced the RACF started-procedures module, initial program load (IPL) the system again with the CLPA option so the new module takes effect. A RACF started-procedure table entry has the following fields:

procname	*useridname*	*groupname*	one byte of flags	eight bytes (reserved)

Note: The last eight bytes are reserved and must be binary zeros.

The started-task name for DFSMShsm is the same as the name of the DFSMShsm startup procedure—DFSMShsm's member name in SYS1.PROCLIB. The name **DFHSM** (as seen in the DFSMShsm starter set "Starter set example" on page 109, and highlighted in Figure 54) is the procedure name that should be added to the RACF-started-procedures table.

```
/****************************************************************/
/*           SAMPLE DFSMSHSM STARTUP PROCEDURE                  */
/****************************************************************/
//HSMPROC   EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT  DD SYSOUT=*
//SYSUT2    DD DSN=SYS1.PROCLIB,DISP=SHR
//SYSIN     DD DATA, DLM='$A'
./ ADD NAME=DFHSM
   .
   .
   .
```

Figure 54. Example of a DFSMShsm Primary Address Space Startup Procedure with a Started-Task Name of DFHSM

Figure 55 is a sample DFSMShsm secondary address space startup procedure for starting aggregate backup and recovery processing (ABARS). Notice the procedure name **DFHSMABR**.

```
//HSMPROC  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSIN    DD DATA,DLM='$A'
./ ADD NAME=DFHSMABR
   .
   .
   .
```

Figure 55. Example of ABARS Secondary Address Space Startup Procedure with a Started-Task Name of DFHSMABR

Table 10 on page 172 lists the table entries for DFSMShsm and ABARS. It shows **DFHSM** as the DFSMShsm procedure name and its associated user ID, **DFHSM**. The ABARS entry reflects **DFHSMABR** as the ABARS procedure name and its associated user ID, **DFHSM**.

Table 10. Example of RACF-Started-Procedure Table Entries for DFSMShsm and ABARS

DFHSM	DFHSM		X'00'	X'0000000000000000'
DFHSMABR	DFHSM		X'00'	X'0000000000000000'

Note: If you are using remote sharing functions (RRSF) to propagate RACF commands to remote hosts, it is recommended that you define the RACF user ID for DFSMShsm with the SPECIAL and OPERATIONS attributes on all of the recipient systems.

Method 2–RACF STARTED class

The second method you can use to associate a user ID with a started task is by using the RACF STARTED class. The RACF STARTED class is a RACF general resource class. The following commands are the equivalent STARTED class entry for the previous **DFHSM** example.

Issue the following commands in order:

1. SETR GENERIC(STARTED)
2. RDEFINE STARTED DFHSM.* STDATA(USER(DFHSM))
3. SETROPTS RACLIST(STARTED) REFRESH

Note: If you are using remote sharing functions (RRSF) to propagate RACF commands to remote hosts, it is recommended that you define the RACF user ID for DFSMShsm with the SPECIAL and OPERATIONS attributes on all of the recipient systems. As an alternative, you can mark DFHSM as trusted in the RACF started procedures table (ICHRIN03) or STARTED class on all of the recipient systems.

Configuring DFSMShsm to invoke DFSMSdss as a started task

When DFSMShsm invokes DFSMSdss using the DFSMSdss cross memory API, DFSMShsm will request that DFSMSdss use a unique address space identifier for each unique DFSMShsm function and host ID. For a list of DFSMSdss address space identifiers, see "DFSMSdss address spaces started by DFSMShsm" on page 381.

If you plan to configure DFSMShsm to start DFSMSdss in its own address space using the DFSMSdss cross memory API, you might need to configure your security system to permit these tasks.

These tasks can be defined explicitly in the same manner as the DFSMShsm started task in "Method 1–RACF started procedures table (ICHRIN03)" on page 170 or they can be defined generically in the same manner as the DFSMShsm started task in "Method 2–RACF STARTED class."

The following is an example of the commands you might use. The user ID DFHSM, also in the example for "Method 2–RACF STARTED class," is used for illustration only.

```
SETR GENERIC(STARTED)
RDEFINE STARTED ARC*.*  STDATA(USER(DFHSM))
SETR RACLIST(STARTED) REFRESH
```

Normally the starting process for DFSMSdss is IEESYSAS. When viewing active jobs in SDSF, IEESYSAS will appear as the job name, and the DFSMShsm passed address space identifier will appear as the step name.

Identifying DFSMShsm to z/OS UNIX System Services

If you plan to use DFSMShsm to backup HFS or zFS data sets mounted by z/OS UNIX System Services, DFSMShsm must have a RACF user ID associated with it, with one of the following levels of authorization:

- For zFS data sets, the RACF user ID associated with DFSMShsm must have UPDATE authority to the SUPERUSER.FILESYS.PFSCTL profile in class UNIXPRIV.
- For HFS data sets, the RACF user ID associated with DFSMShsm must have UPDATE authority to the SUPERUSER.FILESYS.QUIESCE profile in class UNIXPRIV.
- DFSMShsm must be defined to z/OS UNIX System Services as a superuser, and the RACF user ID must have a default RACF group which has an OMVS segment with a group id (GID). The user ID must also have an OMVS segment with the following parameters: UID(0) HOME('/').

Either of these methods provide the required authorization to quiesce or unquiesce a file system. For additional information, refer to *z/OS UNIX System Services Planning*.

Authorizing and protecting DFSMShsm commands in the RACF FACILITY class environment

DFSMShsm provides a way to protect all DFSMShsm command access through the use of RACF FACILITY class profiles. An active RACF FACILITY class establishes the security environment. An active RACF FACILITY class means that DFSMShsm uses RACF protection for all commands instead of using the DFSMShsm AUTH command protection to authorize users to the storage administrator commands.

RACF FACILITY class profiles for DFSMShsm

To use RACF FACILITY class checking, the RACF FACILITY class must be active when DFSMShsm is started. If the RACF FACILITY class is active, the following processing occurs:

- DFSMShsm uses RACF FACILITY class checking for all authorized and user commands.
- DFSMShsm honors profiles in the FACILITY class that are added or modified.
- The ABACKUP and ARECOVER commands are authorized only with the use of RACF FACILITY class.
- Neither the AUTH command nor the UID startup procedure parameter can override the RACF FACILITY class definition.

If the RACF FACILITY class is not active, DFSMShsm uses AUTH or UID to process all storage administrator commands.

Security administrators can use the existing RACF CMDAUTH function to further identify which operator consoles can issue commands to DFSMShsm.

Restriction: If the RACF FACILITY class is active, users can only issue commands to which they are authorized by RACF profiles.

Protecting DFSMShsm commands with RACF FACILITY class profiles

DFSMShsm uses the following set of the RACF FACILITY class profiles to protect commands:

STGADMIN.ARC.*command*
: DFSMShsm storage administrator command protection. This is a profile for a specific DFSMShsm storage administrator command.

STGADMIN.ARC.*command.parameter*
: This profile protects a specific DFSMShsm administrator command with a specific parameter. See Table 11 for the parameters that you can protect.

STGADMIN.ARC.ENDUSER.*h_command*
: DFSMShsm user command protection. This profile protects a specific DFSMShsm user command.

STGADMIN.ARC.ENDUSER.*h_command.parameter*
: This profile protects a specific DFSMShsm user command with a specific parameter. See Table 12 on page 176 for the parameters that you can protect.

Protecting DFSMShsm storage administrator commands with RACF FACILITY class profiles

Security administrators are now responsible for authorizing users and storage administrators to DFSMShsm commands. Each storage administrator command can be protected through the following RACF FACILITY class profile:

- **STGADMIN.ARC.*command***
- **STGADMIN.ARC.*command.parameter***

Storage administrators must have READ access authority to the profile in order to use the command or command and parameter. A security administrator can create the following fully qualified, specific profiles (Table 11) to authorize or deny the use of DFSMShsm storage administrator commands.

Table 11. RACF FACILITY Class Profiles for DFSMShsm Storage Administrator Commands

Command name	RACF FACILITY class resource name
ABACKUP	STGADMIN.ARC.ABACKUP STGADMIN.ARC.ABACKUP.*agname*
ARECOVER	STGADMIN.ARC.ARECOVER STGADMIN.ARC.ARECOVER.*agname* STGADMIN.ARC.ARECOVER.*agname*.REPLACE
ADDVOL	STGADMIN.ARC.ADDVOL
ALTERDS	STGADMIN.ARC.ALTERDS
ALTERPRI	STGADMIN.ARC.ALTERPRI
AUDIT	STGADMIN.ARC.AUDIT
AUTH	STGADMIN.ARC.AUTH
BACKDS	STGADMIN.ARC.BACKDS STGADMIN.ARC.BACKDS.NEWNAME STGADMIN.ARC.BACKDS.RETAINDAYS
BACKVOL	STGADMIN.ARC.BACKVOL
BDELETE	STGADMIN.ARC.BDELETE
CANCEL	STGADMIN.ARC.CANCEL
DEFINE	STGADMIN.ARC.DEFINE
DELETE	STGADMIN.ARC.DELETE
DELVOL	STGADMIN.ARC.DELVOL
DISPLAY	STGADMIN.ARC.DISPLAY

Table 11. RACF FACILITY Class Profiles for DFSMShsm Storage Administrator Commands (continued)

Command name	RACF FACILITY class resource name
EXPIREBV	STGADMIN.ARC.EXPIREBV
FIXCDS	STGADMIN.ARC.FIXCDS
FREEVOL	STGADMIN.ARC.FREEVOL
FRBACKUP	STGADMIN.ARC.FB.*cpname*
FRDELETE	STGADMIN.ARC.FD.*cpname*
FRRECOV	STGADMIN.ARC.FR.*cpname* STGADMIN.ARC.FR.NEWNAME
HOLD	STGADMIN.ARC.HOLD
LIST	STGADMIN.ARC.LIST **2** Exception: STGADMIN.ARC.LC.*cpname*, when COPYPOOL(*cpname*) keyword is specified .
LOG	STGADMIN.ARC.LOG
MIGRATE	STGADMIN.ARC.MIGRATE
PATCH	STGADMIN.ARC.PATCH
QUERY	STGADMIN.ARC.QUERY
RECALL	STGADMIN.ARC.RECALL
RECOVER	STGADMIN.ARC.RECOVER STGADMIN.ARC.RECOVER.NEWNAME
RECYCLE	STGADMIN.ARC.RECYCLE
RELEASE	STGADMIN.ARC.RELEASE
REPORT	STGADMIN.ARC.REPORT
SETMIG	STGADMIN.ARC.SETMIG
SETSYS	STGADMIN.ARC.SETSYS
STOP	STGADMIN.ARC.STOP
SWAPLOG	STGADMIN.ARC.SWAPLOG
TAPECOPY	STGADMIN.ARC.TAPECOPY
TAPEREPL	STGADMIN.ARC.TAPEREPL
TRAP	STGADMIN.ARC.TRAP
UPDATEC	STGADMIN.ARC.UPDATEC
UPDTCDS	STGADMIN.ARC.UPDTCDS

Note:

1. If a storage administrator has access to the AUTH command, their use of it creates, alters, or deletes MCU records. DFSMShsm does not use these MCU records for authorization checking while the FACILITY class is active.
2. The FACILITY class resource name used to protect the LIST COPYPOOL command depends on whether a specific copy pool name is specified in the command. When a copy pool name is not specified, LIST COPYPOOL is protected by the STGADMIN.ARC.LIST resource. When a specific copy pool name is specified, LIST COPYPOOL(*cpname*) is protected by the resource STGADMIN.ARC.LC.*cpname*.

Protecting DFSMShsm user commands with RACF FACILITY class profiles

Each user command can be protected through the following RACF FACILITY class profiles:

- **STGADMIN.ARC.ENDUSER.*h_command***
- **STGADMIN.ARC.ENDUSER.*h_command.parameter***

Users must have READ access authority to the profile in order to use command or command and parameter. A security administrator can create the following fully qualified, specific profiles (Table 12) to authorize or deny the use of DFSMShsm user commands.

Table 12. RACF FACILITY Class Profiles for DFSMShsm User Commands

Command name	**RACF FACILITY class resource name**
HALTERDS	STGADMIN.ARC.ENDUSER.HALTERDS
HBACKDS	STGADMIN.ARC.ENDUSER.HBACKDS STGADMIN.ARC.ENDUSER.HBACKDS.NEWNAME STGADMIN.ARC.ENDUSER.HBACKDS.RETAINDAYS STGADMIN.ARC.ENDUSER.HBACKDS.TARGET
HBDELETE	STGADMIN.ARC.ENDUSER.HBDELETE
HCANCEL	STGADMIN.ARC.ENDUSER.HCANCEL
HDELETE	STGADMIN.ARC.ENDUSER.HDELETE
HLIST	STGADMIN.ARC.ENDUSER.HLIST
HMIGRATE	STGADMIN.ARC.ENDUSER.HMIGRATE
HQUERY	STGADMIN.ARC.ENDUSER.HQUERY
HRECALL	STGADMIN.ARC.ENDUSER.HRECALL
HRECOVER	STGADMIN.ARC.ENDUSER.HRECOVER

Protecting DFSMShsm user macros with RACF FACILITY class profiles

A security administrator can create the following fully qualified discrete profiles (Table 13) to authorize or deny the use of DFSMShsm macro interface commands issued via a macro interface. Additionally, the DFSMShsm implicit processing is detailed in Table 14 on page 177.

Table 13. RACF FACILITY Class Profiles for DFSMShsm User Macros

Macro / interface name	**RACF FACILITY class resource name**
ARCXTRCT	STGADMIN.ARC.ENDUSER.HLIST
ARCHRCLL	STGADMIN.ARC.ENDUSER.HRECALL
ARCFMWE	No protection
ARCHBACK	STGADMIN.ARC.ENDUSER.HBACKDS
ARCHBDEL	STGADMIN.ARC.ENDUSER.HBDELETE
ARCHDEL	STGADMIN.ARC.ENDUSER.HDELETE
ARCHMIG	STGADMIN.ARC.ENDUSER.HMIGRATE
ARCHRCOV	STGADMIN.ARC.ENDUSER.HRECOVER
ARCHSEND	No protection. Instead, the command sent by ARCHSEND is checked by the appropriate profile.

Table 14. RACF FACILITY Class Profiles for DFSMShsm Implicit Processing

Implicit process	RACF FACILITY class resource name
Implicit recall	Protected by user's authority to data set being recalled.
Implicit delete, roll-off	For GDG, protected by user's authority to the data set.

Creating the RACF FACILITY class profiles for ABARS

You can allow all console operators or any user, including a user who is not DFSMShsm authorized, to issue the ABACKUP and ARECOVER commands. This ABARS command authority is controlled by associating the console or user with the RACF FACILITY class profile for ABARS.

ABARS FACILITY classes offer two levels of authorization: *comprehensive* and *restricted*.

Comprehensive authorization allows a user to issue the ABACKUP and ARECOVER commands for all aggregates and DFSMShsm does not check the authority of the user to access each data set in a given aggregate.

Restricted authorization restricts a user to issuing ABACKUP and ARECOVER commands for only the single aggregate specified in the ABARS FACILITY class profile name. DFSMShsm checks the authority of the user to back up each data set in the aggregate that is processed with the ABACKUP command. Because DFSMShsm checks the authority of a user during ABACKUP processing, a user needs at least RACF READ authority to all of the data sets in a given aggregate.

Note:
1. If a user is authorized to both the comprehensive and restricted, that user has restricted authorization and DFSMShsm checks the user's authority for data sets in the specific aggregate. It is generally inadvisable and unnecessary to give a user both comprehensive and restricted authorization.
2. If your installation uses generic profiles then checking is done for the most specific profile first, and if your generic profile grants access to that profile then you have restricted access. See your security administrator or refer to *z/OS Security Server RACF Security Administrator's Guide* for more information.

ABARS comprehensive RACF FACILITY class authorization

You can use the following commands to authorize a person to issue the ABACKUP and ARECOVER commands globally, for all aggregates and DFSMS does not check that persons authority for each data set that is processed.

If you specify . . .	Then . . .
RDEFINE FACILITY STGADMIN.ARC.ABACKUP UACC(NONE) PERMIT STGADMIN.ARC.ABACKUP CLASS(FACILITY) ID(userid) ACCESS(READ)	You define a comprehensive RACF FACILITY class under which the person who is associated with *userid* can issue the ABACKUP command.

If you specify . . .	Then . . .
RDEFINE FACILITY STGADMIN.ARC.ARECOVER UACC(NONE) PERMIT STGADMIN.ARC.ARECOVER CLASS(FACILITY) ID(userid) ACCESS(READ)	You define a comprehensive RACF FACILITY class under which the person who is associated with *userid* can issue the ARECOVER command.

If your installation uses generic profiles then checking is done for the most specific profile first, and if your generic profile grants access to that profile, then you have restricted access. It is suggested that you define STGADMIN.ARC.ABACKUP.* UACC(NONE) for the generic profile and grant users specific access to their aggregates on an individual basis. ARECOVER processing is the same in regards to comprehensive and restrictive resources when processing generic profiles. See your security administrator or refer to *z/OS Security Server RACF Security Administrator's Guide* for more information.

ABARS restricted RACF FACILITY class authorization

The following commands authorize a user to issue the ABACKUP and ARECOVER commands for a single aggregate and DFSMS checks the authorization of the user for each data set processed by the ABACKUP command.

If you specify . . .	Then . . .
RDEFINE FACILITY STGADMIN.ARC.ABACKUP. aggregate_group_name UACC(NONE) and PERMIT STGADMIN.ARC.ABACKUP. aggregate_group_name CLASS(FACILITY) ID(userid) ACCESS(READ)	You define a restricted RACF FACILITY class under which the person who is associated with *userid* can issue the ABACKUP command for the aggregate group specified in the facility class profile name. **Rule:** DFSMShsm checks the authority of the user for each data set in the aggregate. Therefore, a user who is authorized to issue the ABACKUP command for a single aggregate must have at least read authorization for all data sets in the aggregate.
RDEFINE FACILITY STGADMIN.ARC.ARECOVER. aggregate-group-name UACC(NONE) and PERMIT STGADMIN.ARC.ARECOVER. aggregate-group-name CLASS(FACILITY) ID(userid) ACCESS(READ)	You define a restricted RACF FACILITY class under which the user who is associated with *userid* can issue ARECOVER commands for only the aggregate group name associated with *aggregate* when REPLACE was not specified. RACF does not check the authority of the user for recovered data sets.

See *z/OS DFSMShsm Storage Administration* for a discussion of comprehensive and restricted ABACKUP and ARECOVER commands. For more information on RACF FACILITY class profiles, seeo *z/OS Security Server RACF Security Administrator's Guide*.

Creating RACF FACILITY class profiles for concurrent copy

DFSMShsm uses the STGADMIN.ADR.DUMP.CNCURRNT FACILITY class to authorize the use of the concurrent copy options on the data set backup commands. Checking for authorization is done prior to invoking DFSMSdss. If RACF indicates lack of authority, DFSMShsm fails the data set backup request if the concurrent copy request was REQUIRED,VIRTUALREQUIRED or CACHEREQUIRED. If REQUIRED,VIRTUALREQUIRED or CACHEREQUIRED was not specified and RACF indicates a lack of authority, DFSMShsm continues to backup the data set as if the concurrent copy keyword was not specified on the backup command.

Activating the RACF FACILITY class profiles

To set the security environment for DFSMShsm commands, you must activate the RACF FACILITY class before DFSMShsm is started. DFSMShsm uses RACF FACILITY class checking if the RACF FACILITY class is active. If you have not defined the new profiles, every DFSMShsm command fails.

Table 15 lists examples of the RACF commands that provide storage administrators access to a specific storage administrator command and a specific end user command while denying access for other users.

Table 15. Minimum RACF Commands for DFSMShsm

RACF command	Purpose
SETROPTS CLASSACT(FACILITY)	Defines the FACILITY class as active.
SETROPTS RACLIST(FACILITY)	Activates the sharing of in-storage profiles (improves performance).
RDEFINE FACILITY STGADMIN.ARC.command UACC(NONE)	Defines a default, denying all users access to the specific storage administrator command.
PERMIT STGADMIN.ARC.command CLASS(FACILITY) ID(*user1*) ACCESS(READ)	Allows *user1* to issue the specific storage administrator command.
RDEFINE FACILITY STGADMIN.ARC.ENDUSER.command UACC(NONE)	Defines a default, denying all users access to the specific user command.
PERMIT STGADMIN.ARC.ENDUSER.command CLASS(FACILITY) ID(*user1*) ACCESS(READ)	Allows *user1* to issue the specific user command.
SETROPTS RACLIST(FACILITY) REFRESH	Refreshes in-storage profile lists.

Table 16 shows how you can expand the minimal list of RACF commands to further restrict access for storage administrator commands.

Table 16. Expanded RACF Commands for DFSMShsm

RACF command	Purpose
PERMIT STGADMIN.ARC.ADDVOL CLASS(FACILITY) ID(*user2*) ACCESS(READ)	Allows *user2* to issue the ADDVOL command only.
RDEFINE FACILITY STGADMIN.ARC.ENDUSER.HMIGRATE UACC(NONE)	Defines a default, allowing no user access to the HMIGRATE user command.

Authorizing commands issued by an operator

If RACF is active, then MVS uses the z/OS CMDAUTH function to verify that the operator console used to issue the DFSMShsm command is authorized to issue that DFSMShsm command.

If RACF is not active at DFSMShsm startup, then MVS does not perform any verification of commands that are issued from an operator console.

If you stop RACF while DFSMShsm is active, then MVS fails all DFSMShsm commands that are issued from an operator console.

Protecting DFSMShsm resources

Because DFSMShsm manages data set resources on DASD and tape, protection from unauthorized access to DFSMShsm resources is an important consideration.

The following DFSMShsm resources must be protected from unauthorized access:

- DFSMShsm data sets
  - Control data sets
  - Journal
  - Logs
  - Control data set backup versions
  - Small-data-set-packing data sets
  - Migrated data sets
  - Backed up data sets
  - ABARS SYSIN data sets
  - ABARS FILTERDD data sets
  - ABARS RESTART data sets
  - ABARS IDCAMS data sets
- DFSMShsm tapes
  - Level-two (ML2) migration tapes
  - Incremental backup tapes
  - Dump tapes
  - TAPECOPY tapes
  - ABARS tapes

Protecting DFSMShsm data sets

You can protect DFSMShsm data sets with a generic data set profile. At least one generic data set profile must be created of the form '*uid*.**'. **Example:** If you have chosen DFHSM as the high-level qualifier (UID field in "Starter set example" on page 109), use the following commands to create the data set profile:

If you specify . . .	Then . . .
SETROPTS GENERIC(*)	Activates generic profile checking for the DATASET class plus all the classes in the class descriptor table except grouping classes.
ADDGROUP (DFHSM)	Adds the RACF group DFHSM.
ADDSD 'DFHSM.**' UACC(NONE)	RACF creates a generic profile that provides security for all data sets beginning with DFHSM.

After you have created a profile for protecting DFSMShsm data sets, you must authorize some users to access the data sets protected by the profile. A user is authorized to a given profile either by adding the user to the access list or to a group specified in the profile. A user is added to a group with the CONNECT command, see "Authorizing users to access DFSMShsm resources" on page 184 for more information. For more information about generic resources, refer to *z/OS DFSMShsm Storage Administration*.

Protecting DFSMShsm activity logs

If you specify the SETSYS ACTLOGTYPE(DASD) or the SETSYS ABARSACTLOGTYPE(DASD) command, DFSMShsm writes its activity log data to data sets on a DASD volume. DFSMShsm names activity logs with a default high-level-qualifier name of HSMACT.

Example: You can use the following command to protect DFSMShsm activity logs:

If you specify . . .	Then . . .
ADDSD 'HSMACT.**' UACC(NONE)	RACF protects the DFSMShsm activity logs.

Protecting DFSMShsm tapes

You can protect DFSMShsm-managed tapes by:
- Having RACF installed and activated
- Using one of the following options:
  - Defining to RACF the tapes you want to protect by:
    - Defining the RACF TAPEVOL resource class in the RACF descriptor table (CDT).
    - Specifying the SETSYS TAPESECURITY(RACF|RACFINCLUDE) command.

    Tip: If you have not defined RACF tape volume sets for DFSMShsm, but you want RACF to protect all the tapes through a RACF generic profile, specify the SETSYS TAPESECURITY(RACF|RACFINCLUDE) command.
  - Use RACF DATASET class protection using either of the following options:
    - Using RACF SETROPTS TAPEDSN.
    - DEVSUPxx TAPEAUTHDSN=YES

    For more information, see "Protecting tapes" on page 221.

Defining RACF TAPEVOL resource classes

If you choose to use RACF TAPEVOL profiles, the way you define your RACF TAPEVOL resource classes is determined by the number of tapes that you want to protect.

Note: RACF resource names support up to 10000 tape volume sets.

Method 1—Protecting with a single profile: This method defines two RACF resource names for DFSMShsm tapes. One name is for aggregate backup and recovery tapes (HSMABR), and the other name is for all other DFSMShsm tapes (HSMHSM).

If you specify . . .	Then . . .
RDEFINE TAPEVOL HSMABR and RDEFINE TAPEVOL HSMHSM	RACF protects 10000 or fewer tapes.

Method 2—Protecting with multiple profiles: This method defines multiple RACF resource names for DFSMShsm tape volume sets. For method 2 to be properly activated in DFSMShsm, you must activate RACF during DFSMShsm initialization. *In all cases, HSMHSM must be defined.* Aggregate backup and recovery tapes are defined as HSMABR. All other DFSMShsm tapes and are defined by the last nonblank characters that exist as a result of your site's naming conventions for tape volume serial numbers.

If you specify . . .	Then . . .
RDEFINE TAPEVOL HSMHSM RDEFINE TAPEVOL HSMABR RDEFINE TAPEVOL DFHSMA . . . . RDEFINE TAPEVOL DFHSMZ RDEFINE TAPEVOL DFHSM@ RDEFINE TAPEVOL DFHSM$ RDEFINE TAPEVOL DFHSM# RDEFINE TAPEVOL DFHSM- RDEFINE TAPEVOL DFHSM0 . . . . RDEFINE TAPEVOL DFHSM9	RACF protects more than 5000 tapes.

If you elect to choose "Method 2—Protecting with multiple profiles," DFSMShsm associates a tape with a RACF resource name of HSMABR or DFHSMx, where x is the last nonblank character of the tape volume serial number. The set of valid nonblank characters for a tape volume serial number consists of all alphanumeric and national characters and the hyphen, as illustrated by the RDEFINE commands illustrated previously. You need not define any DFHSMx resource names for any x that does not exist as a result of your naming conventions for tape volume serial numbers.

If you specify from each processing unit in a sysplex . . .	Then . . .
RALTER TAPEVOL HSMHSM ADDVOL (DFHSM)	Method 2 is activated in each processing unit from which the command was entered the next time DFSMShsm is initialized in that processing unit. If the SETSYS TAPESECURITY (RACF|RACFINCLUDE) command has been specified, DFSMShsm adds the tapes to the RACF profile, using HSMHSM, HSMABR, or DFHSMx as the resource name. If you do not define the resource name, you will receive errors when DFSMShsm attempts to add tapes to the profile. You should not give DFSMShsm any specific access authorization for HSMHSM, HSMABR, and DFHSMx RACF resource names; however, if you give DFSMShsm specific access authority, the level of authority must be ALTER authority.

Defining the RACF environment to DFSMShsm

You define the RACF environment to DFSMShsm when you specify the SETSYS TAPESECURITY(RACF|RACFINCLUDE) command.

If you specify . . .	Then . . .
SETSYS TAPESECURITY (RACF)	DFSMShsm protects each backup, migration, and dump tape with RACF. If you specify only the RACF security option, DFSMShsm fails the backup or migration of password-protected data sets to tape. **Note:** DFSMShsm does not place a backup version or migration copy of a password-protected data set on a tape that is not password-protected unless you specify the RACFINCLUDE or EXPIRATIONINCLUDE parameters.
SETSYS TAPESECURITY (RACFINCLUDE)	DFSMShsm protects each backup, migration, and dump tape with RACF and DFSMShsm backs up or migrates password-protected data sets to tapes that are not password-protected.

Rule: For Method 2 to be properly activated in DFSMShsm, you must activate RACF and complete RACF definitions prior to DFSMShsm initialization (startup), or you must reinitialize DFSMShsm

Note:

1. The RACF and RACFINCLUDE options are equivalent for dump processing. This is because data sets are not individually processed during volume dump.
2. Converting to a different security method protects only future tapes. Tapes that were protected with a previous security environment retain their original protection.
3. Tapes added to the RACF tape volume set are not initially selected if SETSYS TAPESECURITY(RACF|RACFINCLUDE) is not in effect. Such tapes also are not selected at subsequent selection if the previous tape is not protected by RACF.
4. DFSMShsm can place a RACF-protected backup version of a data set on a backup tape that is not RACF-protected. Similarly, DFSMShsm can place a RACF-protected migration copy of a data set on a migration tape that is not RACF-protected.
5. If no RACF tape volume sets are defined to DFSMShsm but all tapes are protected by an existing RACF generic profile, then specify SETSYS TAPESECURITY(RACF|RACFINCLUDE).

User-protecting tapes with RACF

The system programmer or RACF security administrator can apply RACF protection to tapes before DFSMShsm uses them, except for HSMABR tapes. The ABARS output tapes are both RACF-protected and added to the HSMABR tape volume set if the SETSYS TAPESECURITY(RACF) or SETSYS TAPESECURITY(RACFINCLUDE) options are in effect during aggregate backup.

For the remaining DFSMShsm tapes, if the system programmer or RACF security administrator protects tapes with RACF, the tapes must appear in the RACF tape volume sets for DFSMShsm as follows:

- If you use “Method 1—Protecting with a single profile” on page 181 the volume information must be recorded in the tape volume set of HSMHSM.

If you specify . . .	Then . . .
RALTER TAPEVOL HSMHSM ADDVOL(volser)	RACF protects tapes by adding them to the tape volume set for DFSMShsm. **Note:** DFSMShsm never removes RACF protection from tapes protected with the previous method.

- If you use "Method 2—Protecting with multiple profiles" on page 182, the volume information must be recorded in the tape volume sets of DFHSMx, where x is the last nonblank character of the volume serial number. **Example:** If a tape with a volume serial number of HD0177 is added to DFSMShsm, DFSMShsm associates the volume serial number with the tape volume set of DFHSM7. Likewise, a tape with a volume serial number of T023B is associated with the tape volume set of DFHSMB. To RACF-protect tapes in these instances and to add the tapes to the appropriate tape volume sets for DFSMShsm, use the RALTER command.

If you specify . . .	Then . . .
RALTER TAPEVOL DFHSM7 ADDVOL(HD0177) and RALTER TAPEVOL DFHSMB ADDVOL(T023B)	RACF protects tape volumes HD0177 and T023B.

Tapes already protected in the tape volume set of HSMHSM continue to be protected.

The issuer of these RACF commands must have a certain level of RACF authority. For complete information, refer to *z/OS Security Server RACF Command Language Reference*. If the system programmer or RACF security administrator RACF-protects a tape before DFSMShsm uses it, DFSMShsm never removes the RACF protection from the tape. DFSMShsm removes the RACF protection only from a tape that DFSMShsm has protected with RACF.

Note: These tapes are not selected and used at initial selection if the SETSYS TAPESECURITY(RACF|RACFINCLUDE) parameter is not in effect. Also, these tapes are not selected and used at end-of-volume selection if the previous tape is not RACF-protected.

Protecting scratched DFSMShsm-owned data sets

Some data sets are so sensitive that you must ensure that DASD data, left on a volume after the data set has been scratched from the VTOC, cannot be accessed after the data set is scratched. You can implement this protection when you:

- Specify the SETSYS ERASEONSCRATCH command and
- Specify the ERASE attribute in the RACF profiles for sensitive data sets

If you specify both of the preceding commands, DFSMShsm writes zeros over the sensitive data that is left on the volume after the data set has been scratched from the catalog.

For more information about protecting sensitive data by using the ERASEONSCRATCH technique, refer to *z/OS DFSMShsm Storage Administration*.

Authorizing users to access DFSMShsm resources

Storage administrators who are responsible for migrated data sets and backup versions can be authorized to process catalogs without recalling the migrated data sets and backup versions.

In processing units with both DFSMShsm and RACF active, issuing the UNCATALOG, RECATALOG, or DELETE/NOSCRATCH command against a migrated data set causes DFSMShsm to recall the data set before the operation is performed unless you take action.

To allow certain authorized users to perform these operations on migrated data sets without recalling them, perform the following steps.

1. Define a RACF catalog maintenance group named ARCCATGP.

   Example: ADDGROUP (ARCCATGP)
2. Connect the desired users to that group.

If you specify . . .	Then . . .
CONNECT (*userid1,. . .,useridn*) GROUP(ARCCATGP) AUTHORITY(USE)	Each user (*userid1,. . .,useridn*) is authorized to bypass automatic recall for catalog operations.

Only when such a user is logged on under group ARCCATGP does DFSMShsm bypass the automatic recall for UNCATALOG, RECATALOG, and DELETE/NOSCRATCH requests for migrated data sets.

Example: The following LOGON command demonstrates starting a TSO session under ARCCATGP:

LOGON *userid* | *password* GROUP(ARCCATGP)

Figure 56 demonstrates a batch job running under ARCCATGP:

```
//JOBNAME  JOB (accounting information),'USERNAME',
//         USER=userid,GROUP=ARCCATGP,PASSWORD=password
//         EXEC PGM=....
```

Figure 56. Example of Batch Job Running under ARCCATGP

Note: Automatic recall of a data set being deleted is bypassed *provided that* DFSMShsm receives the DELETE command before any other command against that data set. If another component invokes DFSMShsm prior to the DELETE command, an automatic recall occurs. **Example:** Automatic recall occurs when issuing a DELETE on a migrated data set using TSO option 3.2. A locate is done, which invokes DFSMShsm to locate the data set prior to the DELETE command.

If you use DFSMShsm to back up and recover RLS user catalogs, ensure that the DFSMShsm authorized user identification (UID) has been granted READ access to the IGG.CATLOCK FACILITY class profile and ALTER access to the user catalogs. DFSMShsm invokes DFSMSdss to back up and recover RLS ICF user catalogs. During recover of an RLS user catalog, DFSMShsm specifies the BCSRECOVER(LOCK) parameter, and DFSMSdss will perform LOCK and UNLOCK on behalf of DFSMShsm when the user catalog is preallocated. The RECOVER command will fail if DFSMShsm UID has insufficient authority, or if the IGG.CATLOCK FACILITY class profile has not been defined.

Related reading

For more information on using RLS catalogs and IGG.CATLOCK FACILITY class profile, see *z/OS DFSMS Managing Catalogs*. For more information on the DFSMSdss BCSRECOVER(LOCK|SUSPEND) parameter, see *z/OS DFSMSdss Storage Administration*.

Protecting DFSMShsm commands in a nonsecurity environment

The AUTH command identifies both the DFSMShsm-authorized user who can issue authorized DFSMShsm commands and the DFSMShsm-authorized user who can also add, delete, and change the authority of other DFSMShsm users. When DFSMShsm is installed, the storage administrator with responsibility for DFSMShsm should be identified as the DFSMShsm-authorized user who can affect the authority of other DFSMShsm users.

The AUTH command can be submitted only by users who are already DFSMShsm-authorized users having the database authority control attribute, or the command must be part of the PARMLIB member being processed during DFSMShsm startup.

There is no support at the command level; authorized users have access to all storage administrator (authorized) commands and parameters.

Authorizing and protecting DFSMShsm resources in a nonsecurity environment

If you do not have RACF or similar security software installed, two procedures to submit DFSMShsm-authorized commands in a batch environment without RACF are presented here. Procedure 1, which is the preferred way, allows protection by user ID thus providing better data security. Procedure 2 uses a procedure list that is link edited into an APF-authorized library as an authorized program. One drawback to procedure 2 is that if the procedure name is known by an unauthorized user, data security is lost.

Procedure 1: In this procedure, DFSMShsm is instructed to obtain a user ID from the protected step control block (PSCB) due to the ACCEPTPSCBUSERID parameter of the SETSYS command. It is the installation's responsibility to ensure that a valid user ID is present in the PSCB. See "Determining batch TSO user IDs" on page 84 for more information.

Procedure 2: For this procedure, you can submit operator, storage-administrator, and system-programmer commands for batch processing by defining the HSENDCMD (HSEND) to the Terminal Monitor Program (TMP) as an authorized command and by providing a STEPLIB or JOBLIB card to an Authorized Program Facility (APF) authorized version of module ARCMCMD.

Instead of specifying USER=*userid* on the JOB card, add the HSENDCMD command (HSEND) to the authorized commands table in TMP so that this command can be invoked and submitted to DFSMShsm as an acceptable authorized command.

CSECT IKJEFTE2, within the IKJEFT02 load module, must be modified to indicate that HSENDCMD alias HSEND is an authorized command and should be attached with APF authorization. This modification should be done to the first entry in IKJEFTE2 that contains eight blanks. One blank entry must remain in the authorized command table to indicate the end of the table.

The DFSMShsm module ARCMCMD, which is the HSENDCMD command processor, must be link-edited into an APF-authorized library as an authorized program. The job submitting the HSENDCMD (HSEND) command must use a STEPLIB or JOBLIB card to this library. Access to this APF library must be restricted to prevent unauthorized use of the HSEND command. It is the responsibility of the system programmer to ensure that any DFSMShsm maintenance to module ARCMCMD is also applied to the authorized copy of ARCMCMD. All concatenated STEPLIBs must be authorized. The APF library name must appear either in the system LINKLIST or in the appropriate APF*xx* of the SYS1.PARMLIB. Refer to *z/OS MVS Initialization and Tuning Guide* for additional information about the APF*xx*. Figure 57 shows a sample job that link-edits the ARCMCMD module to create an authorized copy of ARCMCMD.

```
//LINKED   EXEC PGM=IEWL,PARM='LIST,LET,NCAL,XREF,RENT,REUS'
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLMOD  DD  DISP=SHR,DSN=DFHSM.AUTHLIB
//IN       DD  DISP=SHR,DSN=SYS1.CMDLIB
//SYSLIN   DD  *
  INCLUDE IN(HSENDCMD)
  ALIAS HSEND
  SETCODE AC(1)
  ENTRY ARCMCMD
  NAME HSENDCMD(R)
/*
```

Figure 57. Sample Job That Link-Edits the ARCCMCMD Module to Create an Authorized Copy of ARCCMCMD

The successful end of this link-editing results in message IEW0461 for ARCWTU2 and ARCCVT.

Refer to *z/OS TSO/E Customization* for additional information about adding authorized commands to the TSO/E environment. Figure 58 on page 188 is a graphic overview of the DFSMShsm security environment with RACF.

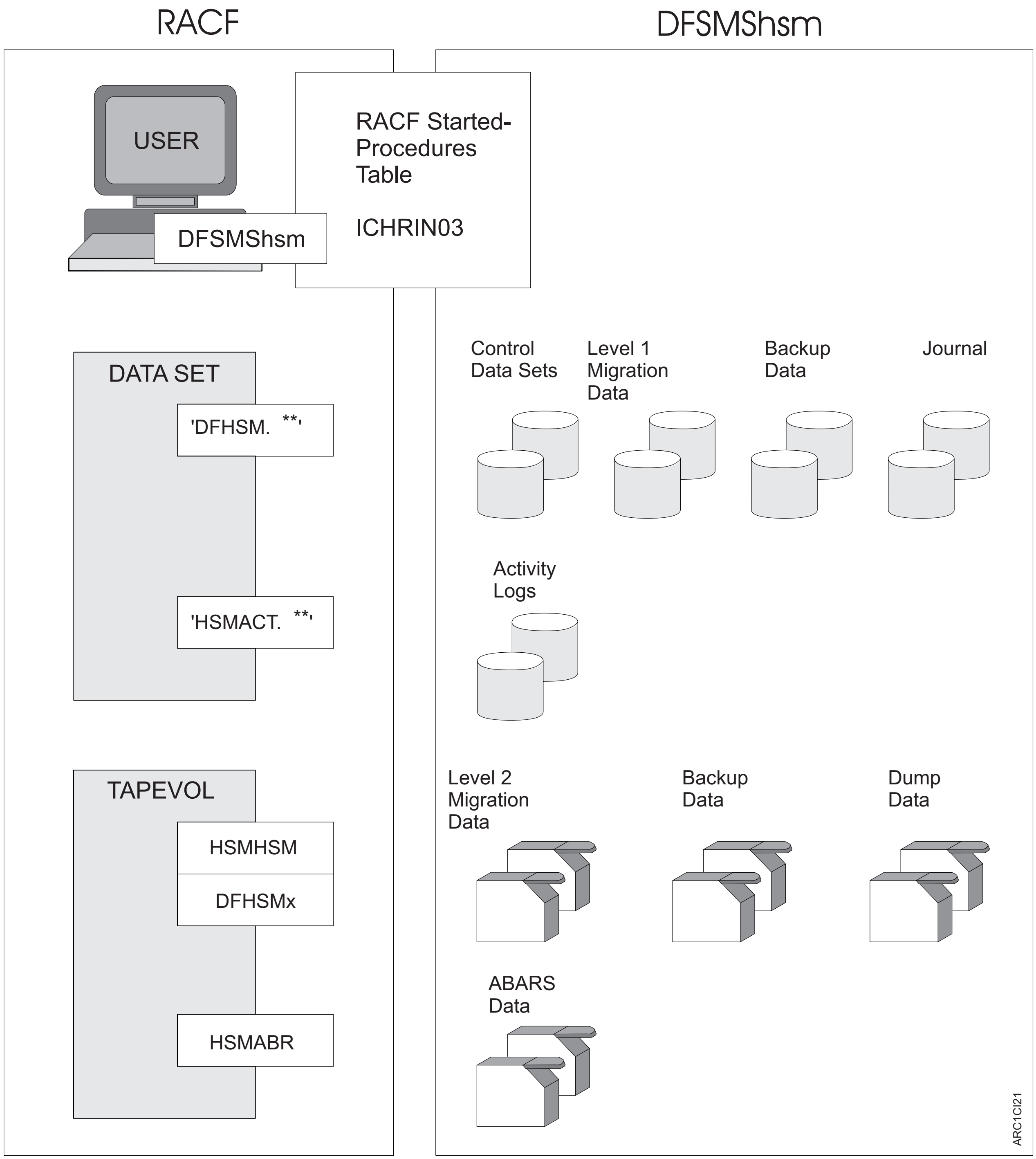


Figure 58. Overview of DFSMShsm with RACF Environment

Chapter 10. Implementing DFSMShsm tape environments

You can implement a DFSMShsm tape environment by specifying SETSYS commands and placing them in the PARMLIB member ARCCMD*xx*. The parameters that you select define your site's tape processing environment.

SETSYS command parameters are different for each site depending on the mix of tape devices, the pools from which these devices select output tapes, and whether the tapes are in an SMS-managed library.

The tape environment, as shown in Figure 59, is defined by the way libraries, tapes, devices, and performance are managed. (The way tapes are managed determines their life cycle as they enter a scratch pool, are selected for output, are inventoried, recycled, and finally returned to a scratch pool.) The way devices are managed determines how they are selected for use, are allocated, and eventually are mounted with tapes. The way you manage performance determines the level of automation and tape utilization at your site.

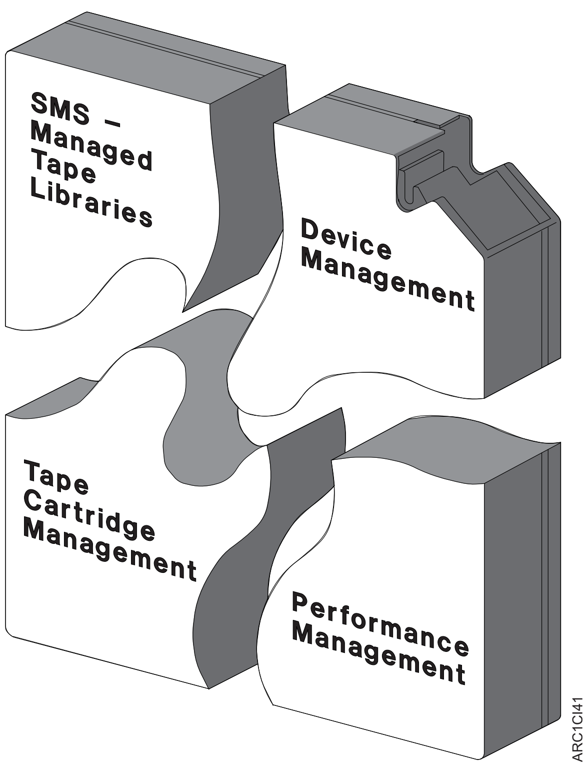


Figure 59. Tape Management Planning Areas

The information in the following topics can help you to understand and then to implement DFSMShsm:

- "Tape device naming conventions" on page 190
- "SMS-managed tape libraries" on page 192
- "Defining the tape management policies for your site" on page 204
- "Implementing the performance management policies for your site" on page 236
- "Initial device selection" on page 247

- “Switching data set backup tapes” on page 251
- “Fast subsequent migration” on page 252

Note: The following discussions relate to SMS-managed tape environments and non-SMS managed tape environments. SMS-managed tape environments refer to environments that support using the SMS Automatic Class Selection routines to direct tape drive allocations, including IBM 3494 and IBM 3495 Tape Library Data Server, and the IBM 3494 Virtual Tape Server. Non-SMS managed tape environments refer to environments that rely on esoteric tape devices to determine allocation, such as stand alone tape drive models, IBM 3494 Tape Library Data Server models managed by Basic Tape Library Support (BTLS), and the non-IBM tape robotic system.

Tape device naming conventions

Table 17 shows the hardware (marketing device name) and software (MVS generic device name) names for tape devices. The MVS generic device names are also the names used by the job control language (JCL) to communicate with the MVS operating system. Because hardware names and software names can be different, we have included Table 17 for your convenience. The following discussions refer to a tape device by its MVS generic device name. Use the MVS generic name for any JCL statements or DFSMShsm commands that specify device names.

Table 17. Tape Device Naming Conventions

Device description	Hardware name	Generic MVS name	Library eligibility
3480 (without compaction capability)	3480	3480	Nonlibrary
3480 (with compaction capability)	3480	3480X	Nonlibrary
3490 (with compaction capability) • 18-track recording • Cartridge-system tape	3490	3480X	Library
3490E • 36-track recording • Cartridge system or enhanced capacity cartridge system tapes	3490	3490	Library
3590B • Serpentine 128-track recording • High performance cartridge or extended high performance cartridge tapes	3590-B1*x*	3590-1 or 3490	Library
3590E • Serpentine 256-track recording • High performance cartridge or extended high performance cartridge tapes • Always in “emulation mode”, either as a 3590B or as a 3490E	3590-E1*x*	3590-1 or 3490	Library
3590H • Serpentine 384-track recording • High performance cartridge or extended high performance cartridge tapes • Always in “emulation mode”, either as a 3590B or as a 3490E device	3590-H1*x*	3590-1 or 3490	Library

Table 17. Tape Device Naming Conventions (continued)

Device description	Hardware name	Generic MVS name	Library eligibility
3592J • Enterprise recording format (EFMT1) • IBM TotalStorage Enterprise tape cartridge (MEDIA5, MEDIA6, MEDIA7, MEDIA8) • Always in “emulation mode” as a 3590B or as a 3490E device (when using MEDIA5 tape)	3592-J1A	3590-1 or 3490	Library
3592-E05 • Enterprise recording format EFMT1 or EFMT2 for MEDIA5, MEDIA6, MEDIA7, or MEDIA8. EFMT2 is required for MEDIA9 and MEDIA10 • IBM TotalStorage Enterprise tape cartridge (MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9, MEDIA10) • Always in “emulation mode” as a 3590B device	3592-E05	3590-1	Library
3592-E06 • Enterprise recording format EFMT1 (read only), EFMT2, EEFMT2, EFMT3 or EEFMT3 for MEDIA5, MEDIA6, MEDIA7 or MEDIA8. EFMT2, EEFMT2, EFMT3 or EEFMT3 is required for MEDIA9 and MEDIA10 • IBM TotalStorage Enterprise tape cartridge (MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9, MEDIA10) • Always in “emulation mode” as a 3590B device	3592-E06	3590-1	Library

3592-E07
• Reads EFMT1 on MEDIA5, MEDIA6, MEDIA7, and MEDIA8 cartridges. Reads EFMT2, EEFMT2, EFMT3, and EEFMT3 on MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9, and MEDIA10 cartridges. Writes EFMT3 and EEFMT3 on MEDIA9 and MEDIA10 cartridges. Reads and writes EFMT4 and EEFMT4 on MEDIA11, MEDIA12, and MEDIA13 cartridges.
• IBM Enterprise tape cartridge:
  – Read only: MEDIA5, MEDIA6, MEDIA7, MEDIA8
  – Read/Write: MEDIA9, MEDIA10, MEDIA11, MEDIA12, MEDIA13
• Always in “emulation mode” as a 3590B device
	3592-E07	3590-1	Library

SMS-managed tape libraries

This section deals exclusively with issues that arise when implementing an SMS-managed tape library. A first consideration for implementing the DFSMShsm tape processing environment is whether any of your tapes are SMS-managed in tape libraries. Tape library configurations are defined with both SMS constructs and DFSMShsm commands. Nonlibrary environments are defined with only DFSMShsm commands (no SMS constructs) and require the use of esoteric unit names. Both library and nonlibrary environments can include a security program and a tape management program.

An SMS-managed tape library is a named collection of storage groups, tape devices, tape cartridges, and tape library dataservers. Libraries using the IBM 3494 or 3495 Tape Library Dataservers provide unattended operation and offer maximum automation of a tape environment. By defining a storage class and one or more storage groups, you can associate specific tapes with a tape library. Storage groups can also span libraries.

| **Note:** A global scratch tape pool must be used if multiple tape libraries are
| assigned to the same storage group.

Implementing an SMS-managed tape library requires that you perform the following tasks:
- "Steps for defining an SMS-managed tape library"
- "Converting to an SMS-managed tape library environment" on page 197
- "Introducing tape processing functions to the library" on page 199

For information about the SMS constructs required to implement tape libraries, refer to *z/OS DFSMS Implementing System-Managed Storage*. For information about the DFSMShsm commands that create tape environments, see "Defining the tape management policies for your site" on page 204.

Steps for defining an SMS-managed tape library

Perform the following steps to implement an SMS-managed tape library:
1. Determine which tape functions that you want to process in a tape library.
2. Set up a global scratch pool.
3. Define or update a storage class to enable a storage group.
4. Define or update a data class to compact tape library data.
5. Define or update a storage group to associate tape devices with the library.
6. Set up or update ACS routines to filter data sets to the library.
7. Define or update the DFSMShsm tape environment in the ARCCMD*xx* PARMLIB member.

Requirement: For DFSMShsm processing to complete successfully in a tape library environment, the OAM address space must have at least been started since the last IPL. This requirement exists at initial IPL, as well as after each subsequent IPL.

Determine which functions to process in a tape library

Tape libraries can process any DFSMShsm tape functions. You must decide which DFSMShsm functions to process in a tape library. Each DFSMShsm function uses a unique data set name. An ACS routine can recognize the functions you want to process in an SMS-managed tape library by the data set names. The data set names are shown in Table 19 on page 194.

Set up a global scratch pool

Set up (as discussed in "Obtaining empty tapes from scratch pools" on page 207) a global scratch pool from which the tape library can obtain scratch tapes. Global scratch pools are recommended for SMS-managed tape libraries because global scratch pools enable maximum automation and use of a tape management product (for example, DFSMSrmm). Using a global scratch pool will also facilitate assigning more than one tape library to a storage group.

Define a storage class

Set up one storage class for DFSMShsm functions you want to implement in a tape library. The storage class enables the device storage groups in an SMS-managed tape library. Refer to *z/OS DFSMSdfp Storage Administration* for more information about storage classes.

Define a data class

Define a data class that restricts device selection to tape devices that support tape processing productivity technologies such as hardware compaction, capacity cartridge system tape type, and recording technology. Table 18 shows the data class attributes that support these performance enhancements. For more information about data classes, refer to *z/OS DFSMSdfp Storage Administration*.

Table 18. Data Class Attributes

Data Class Attribute	Determines
COMPACTION	Whether hardware compaction algorithms compact tapes.
MEDIA TYPE	Whether DFSMShsm writes to standard or enhanced capacity cartridge system tapes. The following media types are valid: • MEDIA1—3490 standard • MEDIA2—3490 enhanced • MEDIA3—3590 standard • MEDIA4—3590 enhanced • MEDIA5—standard, 5 media types • MEDIA6 • MEDIA7 • MEDIA8 • MEDIA9 • MEDIA10 • MEDIA11 • MEDIA12 • MEDIA13
RECORDING TECHNOLOGY	Whether 18-track, 36-track, 128-track, 256-track, 384-track, EFMT1, EFMT2, EFMT3, EFMT4, EEFMT2, EEFMT3, or EEFMT4 recording is used.

Note: Since Open/Close/EOV (during EOV processing) maintains the same recording format across all volumes of a multi-volume data set, customers may also want to mark their previous technology volumes full to force any new data being written to a new tape volume and the new tape technology specified in the data class.

Define a storage croup

Set up at least one storage group, more if you want to associate different storage groups with each DFSMShsm function. For more information about defining storage groups, refer to *z/OS DFSMSdfp Storage Administration*.

Set up the ACS routines

Set up automatic class selection (ACS) routines as the single point of library control for filtering data sets to library resident tapes and devices. When DFSMShsm requests a tape device allocation from the z/OS operating system, ACS routines are invoked and the tape data set name, unit type, and DFSMShsm started task job name are passed to the routines. This information is used by the ACS routines to provide a storage class and a tape storage group that are associated with a tape library.

Tip: Use duplex processing to route original and alternate tapes to different locations simultaneously.

For more information about setting up ACS routines, refer to *z/OS DFSMSdfp Storage Administration*. For more information about duplex processing and ACS routines, see "Creating concurrent tapes for on-site and offsite storage" on page 245.

Information passed to an ACS routine: The information in Table 19 can be used to make filtering decisions in the ACS routines.

Note: If *unittype* is not specified on each of the following commands, it will not be provided as input when DFSMShsm invokes the ACS routines for that function. If not specified, *unittype* often defaults to another setting or generic device, such as 3590-1.

Table 19. DFSMShsm Tape Data Set Names and Unittypes Passed to the ACS Routine

DFSMShsm Function	DFSMShsm Tape Data Set Names	Commands with *unittype* Restrictions
Backup to original	*prefix*.BACKTAPE.DATASET	SETSYS BACKUP(TAPE(*unittype*))
Backup to alternate	*prefix*.copy.BACKTAPE.DATASET	
Recycle of backup tapes to original	*prefix*.BACKTAPE.DATASET	SETSYS - RECYCLEOUTPUT(BACKUP(*unittype*))
Recycle of backup tapes to alternate	*prefix*.copy.BACKTAPE.DATASET	
Migration to original	*prefix*.HMIGTAPE.DATASET	SETSYS TAPEMIGRATION(- DIRECT(TAPE(*unittype*)) | ML2TAPE(TAPE(*unittype*)) | NONE(ROUTETOTAPE(*unittype*)))
Migration to alternate	*prefix*.copy.HMIGTAPE.DATASET	
Recycle of migration tapes to original	*prefix*.HMIGTAPE.DATASET	SETSYS - RECYCLEOUTPUT (MIGRATION(*unittype*))
Recycle of migration tapes to alternate	*prefix*.copy.HMIGTAPE.DATASET	
Dump	*prefix*.DMP.*dclass*.V*volser*.D*yyddd*.T*ssmmhh*	DEFINE DUMPCLASS(*class* UNIT(*unittype*))
Spill	*prefix*.BACKTAPE.DATASET	SETSYS SPILL(TAPE(*unittype*))

Table 19. DFSMShsm Tape Data Set Names and Unittypes Passed to the ACS Routine (continued)

DFSMShsm Function	DFSMShsm Tape Data Set Names	Commands with *unittype* Restrictions
Tape copy (backup tapes)	*prefix*.COPY.BACKTAPE.DATASET	TAPECOPY ALTERNATEUNITNAME (*unittype1*, *unittype2*) TAPECOPY ALTERNATE3590UNITNAME (*unittype1*, *unittype2*)
Tape copy (migration tapes)	*prefix*.COPY.HMIGTAPE.DATASET	
CDS backup DATAMOVER=HSM	*uid*.BCDS.BACKUP.*Vnnnnnnn* *uid*.MCDS.BACKUP.*Vnnnnnnn* *uid*.OCDS.BACKUP.*Vnnnnnnn* *uid*.JRNL.BACKUP.*Vnnnnnnn*	SETSYS CDSVERSIONBACKUP (BACKUPDEVICECATEGORY - (TAPE UNITNAME(*unittype*)))
CDS backup DATAMOVER=DSS	*uid*.BCDS.BACKUP.*Dnnnnnnn* *uid*.MCDS.BACKUP.*Dnnnnnnn* *uid*.OCDS.BACKUP.*Dnnnnnnn* *uid*.JRNL.BACKUP.*Dnnnnnnn*	
ABARS processing for: -control file -DSS data file -instruction file -internal data file	 *outputdatasetprefix*.C.*CccVnnnn* *outputdatasetprefix*.D.*CccVnnnn* *outputdatasetprefix*.I.*CccVnnnn* *outputdatasetprefix*.O.*CccVnnnn*	ABACKUP *agname* UNIT(*unittype*)

Define the DFSMShsm tape library environment in the ARCCMDxx PARMLIB member

Define the SMS-managed tape library environment to DFSMShsm by specifying (in the ARCCMD*xx* PARMLIB member) the DFSMShsm commands that work with the SMS constructs to define a tape library. SMS-managed tape libraries manage physical tape cartridges. DFSMShsm manages the data on those cartridges. For more information about defining the ARCCMD*xx* PARMLIB member, see “Parameter libraries (PARMLIB)” on page 297.

Example: Defining a DFSMShsm environment for SMS-managed tape libraries

For an SMS-managed tape library, many of the management policies for tapes and devices are determined by SMS constructs. These SMS constructs override the *unittype* parameter of the DFSMShsm commands that control non-SMS tapes and devices. The following table shows the DFSMShsm commands that control tape or device policy and the SMS constructs that override the DFSMShsm commands when tapes are processed in an SMS-managed tape library.

The following commands are simplified because you do not need to provide a unit type with the *unittype* parameter if you are filtering on a data set name. The storage class and storage group direct a request to a library and the data class controls the device selection.

Function	SMS Construct	DFSMShsm Command
Compact data on tapes	Data class COMPACTION attribute	SETSYS TAPEHARDWARECOMPACT | NOTAPEHARDWARECOMPACT

Function	SMS Construct	DFSMShsm Command
Restrict output to specific devices	Data class MEDIA TYPE attribute and Data class RECORDING TECHNOLOGY attribute	ABACKUP *agname* UNIT(*unittype*) ARECOVER *agname* TARGETUNIT(*unittype*) DEFINE DUMPCLASS(*class*...UNIT(*unittype*)) SETSYS ABARSUNITNAME(*unittype*) ARECOVERML2UNITNAME(*unittype*) ARECOVERUNITNAME(*unittype*) BACKUP(TAPE(*unittype*)) CDSVERSIONBACKUP (BACKUPDEVICECATEGORY(TAPE (UNITNAME(*unittype*)))) MIGUNITNAME(*unittype*) RECYCLEOUTPUT(BACKUP(*unittype*)) RECYCLEOUTPUT(MIGRATION(*unittype*)) SPILL(TAPE(*unittype*)) TAPEMIGRATION(DIRECT(TAPE(*unittype*))) TAPEMIGRATION(NONE(ROUTETOTAPE(*unittype*))) TAPEMIGRATION(ML2TAPE(TAPE(*unittype*))) UNITNAME(*unittype*))) TAPECOPY ALTERNATEUNITNAME(*unittype1*, *unittype2*) TAPEREPL ALTERNATEUNITNAME(*unittype*) ALTERNATE3590UNITNAME(*unittype1*, *unittype2*)

Figure 60 is an example of the SETSYS commands that define a typical automated tape library (ATL) environment.

```
/***********************************************************************/
/*  SETSYS COMMANDS IN THE ARCCMDXX PARMLIB MEMBER THAT DEFINE THE     */
/*  DFSMSHSM ENVIRONMENT FOR AN SMS-MANAGED TAPE LIBRARY.              */
/***********************************************************************/
/*
SETSYS DUPLEX(BACKUP MIGRATION)
SETSYS SELECTVOLUME(SCRATCH)
SETSYS PARTIALTAPE(MARKFULL)
SETSYS TAPEDELETION(SCRATCHTAPE)
SETSYS BACKUP(TAPE)
SETSYS TAPEMIGRATION(ML2TAPE)
SETSYS TAPESECURITY(RACFINCLUDE EXPIRATIONINCLUDE)
DEFINE DUMPCLASS(ATLHSM -
       NORESET AUTOREUSE NODATASETRESTORE -
       DISPOSITION('AUTOMATE LOCATION'))
/*
```

Figure 60. Sample Automated Tape Library Environment Definition

SETSYS DUPLEX

Duplex processing provides an alternative to TAPECOPY processing for backup and migration of *cartridge* tapes. Duplex processing creates two tapes concurrently; the original tape may be kept onsite while the alternate tape may be either taken offsite or written to a remote tape library. See “Creating concurrent tapes for on-site and offsite storage” on page 245 for more information about the DUPLEX keyword.

SETSYS SELECTVOLUME(SCRATCH)

Specifying that DFSMShsm select scratch tapes as the initial tape for dump and as the subsequent tape for dump, migration, and backup is nearly always recommended for SMS-managed tape libraries because libraries are most efficient when they perform nonspecific mounts. See "Global scratch pools" on page 207 for more information about nonspecific (global) scratch pools.

SETSYS PARTIALTAPE(MARKFULL)
Migration and backup tapes that are partially filled during tape output processing are marked full. This enables a scratch tape to be selected the next time the same function begins. Marking tapes full enables full exploitation of the cartridge loaders because the cartridge loaders can be filled with scratch tapes between tape processing windows. MARKFULL should be used in a virtual tape environment to improve performance and improve backstore tape utilization.

When the total tape-media use and reducing recycle overhead are more important than cartridge-loader exploitation, PARTIALTAPE(REUSE) can be specified. In a REUSE environment, tapes are fully utilized and the amount of recycle processing is reduced. For more information about the PARTIALTAPE parameter, see "Selecting a scratch pool environment" on page 211.

Because a request to recall a data set can "take away" a migration volume that is currently associated with a migration task, such a partial migration tape is created as if you had specified PARTIALTAPE(REUSE).

SETSYS TAPEDELETION(SCRATCHTAPE)
The SCRATCHTAPE option tells DFSMShsm that recycled migration and backup tapes, along with expired dump tapes, are to be returned to a global scratch pool. You should specify TAPEDELETION(SCRATCHTAPE) when a global scratch pool is in use. For more information about specifying what to do with empty tapes, see "Selecting a scratch pool environment" on page 211.

SETSYS BACKUP(TAPE)
Backup processing is to tape devices.

SETSYS TAPEMIGRATION(ML2TAPE)
The level-2 migration medium is tape. SMS data is directed to level 2 according to its management class. Non-SMS data normally moves to level-1 DASD before migrating to level 2.

SETSYS TAPESECURITY(RACFINCLUDE EXPIRATIONINCLUDE)
Because most sites use a tape management program and because most tape management programs are controlled by expiration dates, many sites require multiple security options to protect tapes. DFSMSrmm does not require the expiration option. For more information about protecting tapes, see "Protecting tapes" on page 221. For more information about DFSMSrmm, refer to *z/OS DFSMSrmm Implementation and Customization Guide* and "Using DFSMShsm and DFSMSrmm" in *z/OS DFSMShsm Storage Administration*.

DEFINE DUMPCLASS
The AUTOREUSE option of the DEFINE DUMPCLASS command is especially useful for tape libraries; when dump tapes expire and those dump tapes are associated with a dump class of AUTOREUSE, the tapes are immediately returned to the scratch pool for reuse. If NOAUTOREUSE had been specified, you would have to explicitly DELVOL dump tapes before you could reuse them.

Converting to an SMS-managed tape library environment

Converting to an SMS-managed tape library requires that you consider how you will perform the following tasks:

- Insert tapes into the library
- Identify tapes in and out of the library

- Introduce tape functions to the tape library environment

Inserting DFSMShsm tapes into a tape library

The tapes that you insert into an SMS-managed tape library either contain valid data or are empty.

Tapes containing valid data should be assigned a private status, they should be associated with a tape storage group, and they should be assigned the correct recording technology, compaction, and media type.

Empty tapes not defined to DFSMShsm should be assigned a scratch status and associated with the global scratch pool. This is required if multiple tape libraries are assigned to the same storage group.

Empty tapes defined to DFSMShsm (specific scratch pool) should be assigned a private status and they should be associated with the tape storage group for the DFSMShsm function to which the tapes are ADDVOLed. Once a storage group is assigned to a tape volume, the storage group cannot be changed except with ISMF, JCL, or by returning the tape to scratch status.

Identifying tape library tapes

Because DFSMShsm functions that process tapes in a tape library require the complete connected set of tapes to be in the library, it is helpful to be able to identify both individual tapes and connected sets in a tape library. Enhancements to the DFSMShsm LIST command allow you to identify the status of individual tapes and connected sets.

Tape connected sets: A backed up or migrated data set rarely ends at the exact physical end of a tape. Data sets often span tapes because as a data set is copied, the tape is filled, another tape is requested, and any remaining data is copied to the next tape. Because the data set now spans these two tapes, these tapes are logically related.

A group of DFSMShsm tapes that are logically related is called a connected set. For backup and migration tapes, this relationship exists because data sets can span tapes. For dump tapes, this relationship exists because the tapes are part of a dump copy. Because connected sets can be large in size, DFSMShsm provides a method of reducing the size of a connected set by reducing occurrences of data sets that span physical tape volumes. The creation of data sets that span tapes in a connected set (spanning data sets) can be reduced with the SETSYS TAPESPANSIZE command, which is discussed in detail in *z/OS DFSMShsm Storage Administration*.

Looking at your tape library with the LIST command: Output from the LIST commands indicates whether:

- Connected sets contain data sets that span multiple tapes
- Tapes are partially full, full, or empty
- Backup and migration tapes are managed in an SMS-managed tape library
- Dump tapes are managed in an SMS-managed tape library
- A tape is marked full and has an alternate volume

Table 20 on page 199 shows the LIST commands that are helpful for working with a tape library. The output of the LIST command includes the library name and the storage group or an indication that the tape is not a member of any library. For more information about the LIST command, see *z/OS DFSMShsm Storage Administration*.

Table 20. Library Usage of LIST Command

Requirement	LIST command
All tapes in a connected set must be inserted together and must be ejected together`.	**LIST TTOC SELECT(CONNECTED)** Identifies backup and migration data sets that span backup and migration tapes **LIST TTOC SELECT(NOTCONNECTED)** Identifies backup and migration data sets that do not span backup and migration tapes
The selection status of tapes is that they are either full, not full, or empty.	**LIST TTOC SELECT(FULL)** Identifies tapes that are full **LIST TTOC SELECT(NOTFULL)** Identifies tapes that are not full **LIST TTOC SELECT(EMPTY)** Identify tapes that are empty
The library status of tapes is that they are in a library or not in any library.	**LIST TTOC SELECT(LIB)** Identifies tapes that are in a library **LIST TTOC SELECT(NOLIB)** Identifies tapes that are not in any library
The library status of dump tapes is that they are in a library or not in any library.	**LIST DUMPVOLUME SELECT(LIB)** Identifies dump tapes that are in a library **LIST DUMPVOLUME SELECT(NOLIB)** Identify dump tapes that are not in any library
TAPECOPY or duplex alternate tapes are often ejected from a library and moved to another physical site. These tapes may be written to a tape library that is located on or offsite.	**LIST TTOC SELECT(ALTERNATEVOLUME)** Identifies tapes that have an alternate volume **LIST TTOC SELECT(DISASTERALTERNATEVOLUME)** Identifies tapes that are disaster alternate volumes
The selection status of tapes is that they are full as a result of an error on alternate tape.	**LIST TTOC SELECT(ERRORALTERNATE)** Identifies tapes that are marked full prematurely
The library status of tapes that are part of an aggregate. Tapes associated with an aggregate are often ejected from a library and moved to another site.	**LIST AGGREGATE(*agname*)** Identifies tapes within an aggregate and identifies the library status of those tapes

Introducing tape processing functions to the library

When considering which functions to process in a tape library, think in terms of space-management processing, availability-management processing, or control data set backup processing as indicated in the following implementation scenarios.

As you review these scenarios, notice that the migration and backup conversions are implemented differently. The implementations contrast each other because, for migration, existing tapes are inserted into the library and for backup, scratch tapes are inserted into the library and the existing backup tapes are left in shelf storage. There is no requirement to convert in this manner, but the different methods are shown as examples of the different ways to convert to an SMS-managed tape library.

The DFSMShsm ARCTEEXT exit supports SMS tape library management so that not all tapes for a given DFSMShsm function are required to be located either inside or outside a library. The ARCTEEXT (Tape Ejected) exit provides the capability of dynamically inserting nonlibrary-resident DFSMShsm migration, backup, and dump tapes as they are needed for input. This capability enables a customer to use a smaller automatic library and still have the choice of either allocating tape volumes outside the library or dynamically reinserting tapes whenever ejected tapes are needed. This action can be done without failing jobs or adversely affecting other DFSMShsm activity. The ARCTEEXT exit gains control at a point before code supporting the TAPEINPUTPROMPT function is processed as well as before tape allocation is required.

Scenario 1–Implementing migration processing in an automated tape library

Figure 61 on page 201 is an overview of the conversion environment for migration processing. The following steps are recommended for converting to an automated tape library:

1. Fill the library with existing ML2 and scratch tapes. The LIST ML2 command can be used to identify ML2 tapes that are outside the library. For more information about the LIST command, see *z/OS DFSMShsm Storage Administration*.
2. Set up the ACS routines to map all new migrations to the library. In order to create original and alternate tapes in the same or different locations, ACS routines must filter on both data set names. For information on how to set up ACS routines, refer to *z/OS DFSMS Implementing System-Managed Storage*.
3. All mounts for recalls from library-resident tapes will be automated. If a DFSMShsm migration tape has been ejected from the library, the ARCTEEXT installation exit can be used to enable the capability to dynamically insert the tape whenever it is needed.
4. Eject alternate volumes if they are not already outside the library. These tape volumes may have been created as a result of duplex or TAPECOPY processing. You can identify them by using the LIST TTOC SELECT (ML2 FULL LIB(ALT)) command.

   Note: If your remote locations are creating tapes simultaneously, you may even have an entirely different set of tape management policies.

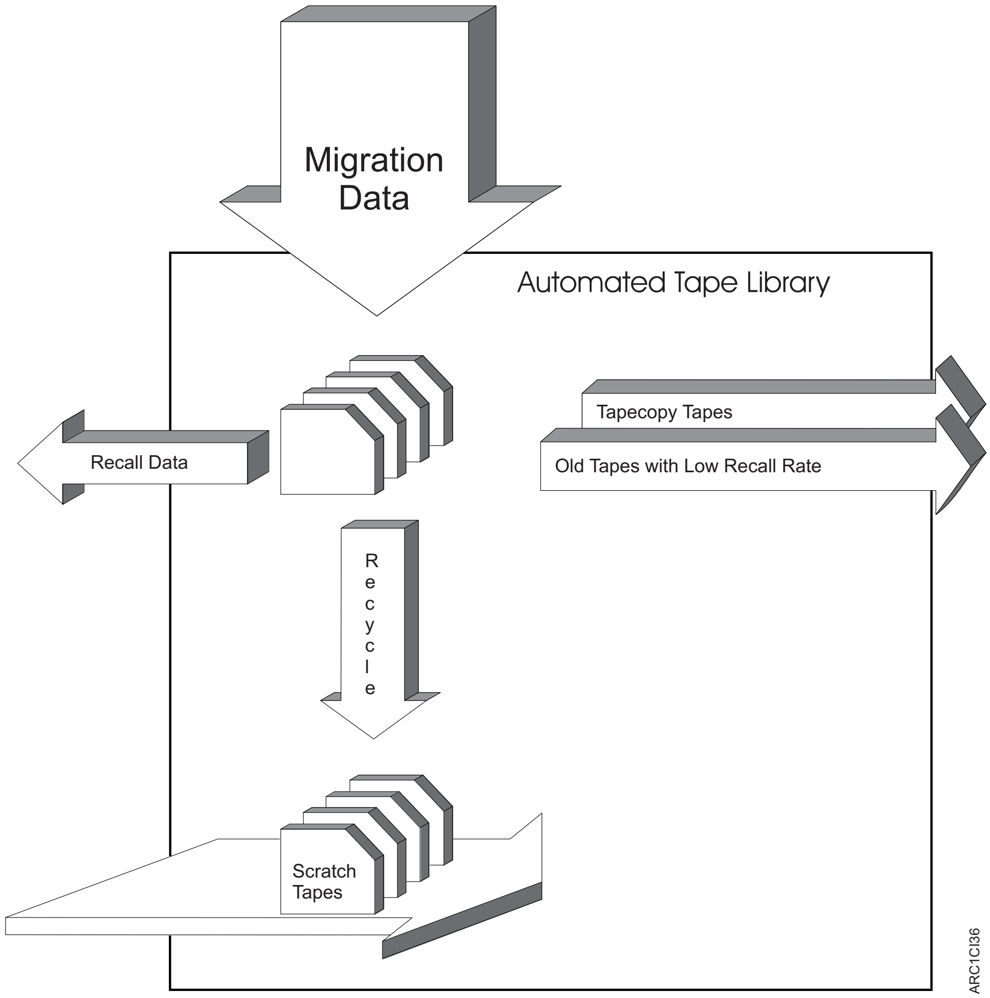


Figure 61. Overview of Implementing Migration in an Automated Tape Library

Implementing migration processing in a manual tape library

When you are converting to a manual tape library (MTL), you need to set up the ACS routines to map all new migrations to the library. In order to create original and alternate tapes in the same or different locations, ACS routines must filter on both data set names. For information on how to set up ACS routines, refer to *z/OS DFSMS Implementing System-Managed Storage*.

Scenario 2–Implementing backup processing in an automated tape library

Figure 62 on page 202 is an overview of the conversion environment for backup processing. The following steps are recommended for converting to an automated tape library (ATL):

1. Fill the library with scratch tapes.
2. Set up the ACS routines to map all new backups to the library. For information on how to set up ACS routines, refer to *z/OS DFSMS Implementing System-Managed Storage*.
3. Mount the tapes. Mounts for recovery from previous backups will be manual but will phase out as more backup processing occurs within the automated tape library. Mounts for recovery from library processing occur within the ATL.

4. Eject connected sets if additional library space is required. If a DFSMShsm backup connected set has been ejected from the library, the ARCTEEXT installation exit can be used to dynamically insert tapes whenever they are needed.
5. Eject alternate volumes if they are not already outside the library. These tape volumes have been created as a result of TAPECOPY processing, and you can identify them by using the LIST TTOC SELECT (BACKUP FULL LIB(ALT)) command.

   Note: If your remote locations are creating tapes simultaneously, you may even have an entirely different set of tape management policies.

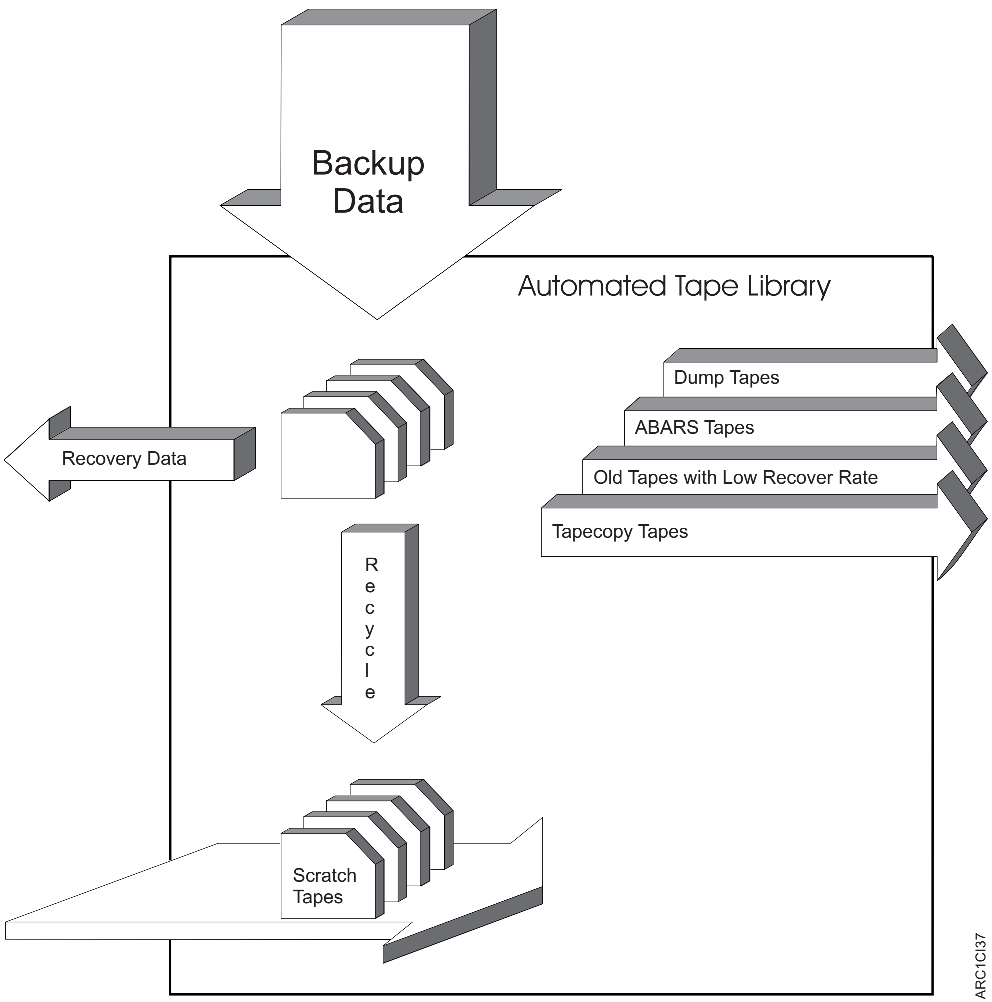


Figure 62. Overview of Implementing Backup in an Automated Tape Library

Implementing backup processing in a manual tape library

The following steps are recommended for converting to a manual tape library (MTL):

1. Set up the ACS routines to map all new backups to the library. For information on how to set up ACS routines, refer to *z/OS DFSMS Implementing System-Managed Storage*.
2. Mount the tapes. Mounts for recovery from previous backups will be manual.

Scenario 3–Implementing control data set backup in an automated tape library

Figure 63 is an overview of the conversion environment for control data set backup processing. The following steps are recommended for converting to an ATL:

1. Fill the library with scratch tapes.
2. Set up the ACS routines to map control data set backup versions to the library. In order to create original and alternate tapes in the same or different locations, ACS routines must filter on both data set names. For information on how to set up ACS routines, refer to *z/OS DFSMS Implementing System-Managed Storage*.
3. CDS backup versions never need to be ejected because old versions roll off and become scratch tapes as they are replaced with current versions.

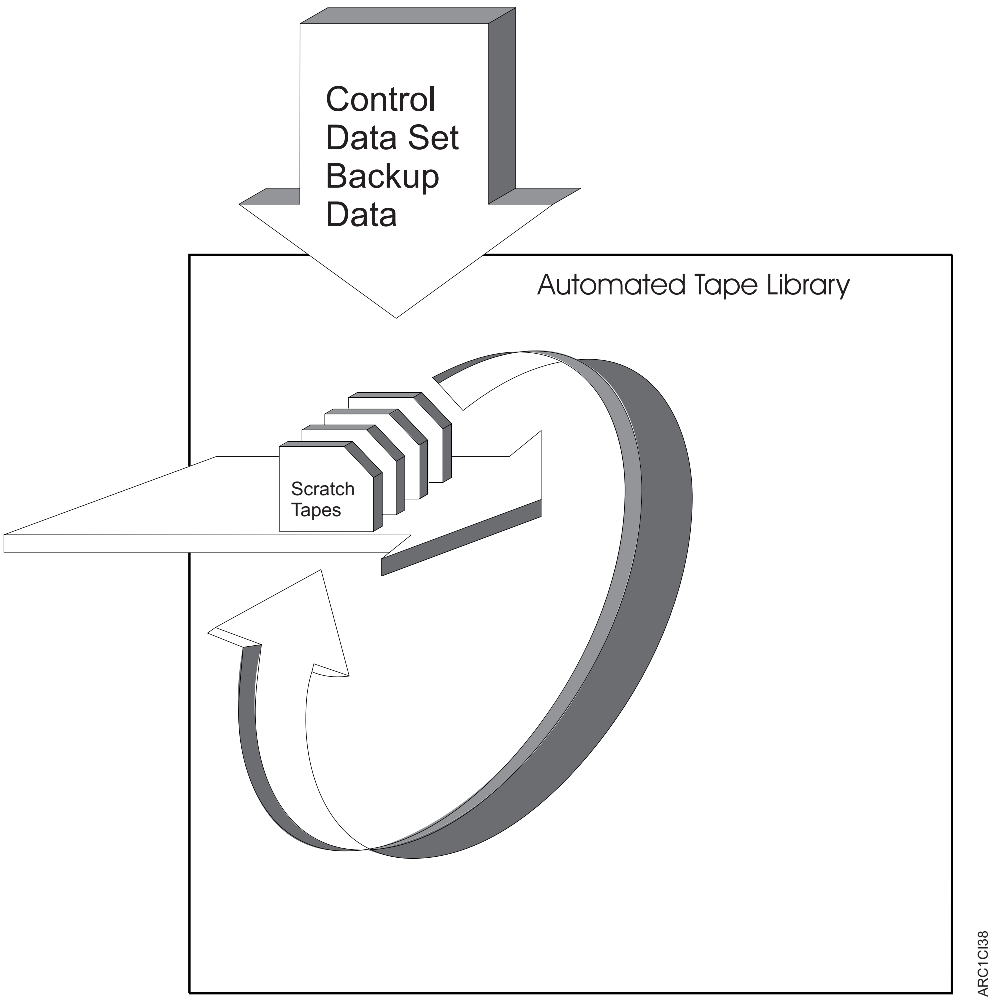


Figure 63. Overview of Implementing Control Data Set Backup in an Automated Tape Library

Implementing control data set backup in a manual tape library

When you are converting to a manual tape library (MTL), you need to set up the ACS routines to map control data set backup versions to the library. In order to create original and alternate tapes in the same or different locations, ACS routines must filter on both data set names. For information on how to set up ACS routines, see *z/OS DFSMS Implementing System-Managed Storage*.

Defining the tape management policies for your site

The tape management policies at your site determine how a tape is managed through its life cycle as it enters a scratch pool, is selected for output, is inventoried as active data, is recycled, and finally is returned to the scratch pool. Figure 64 on page 205 illustrates the life cycle of a tape. Your site's tape management policy is determined by the choices you make about the following tasks:

- "DFSMShsm tape media" on page 206
- "Obtaining empty tapes from scratch pools" on page 207
- "Selecting output tape" on page 209
- "Selecting a scratch pool environment" on page 211
- "Implementing a recycle schedule for backup and migration tapes" on page 216
- "Returning empty tapes to the scratch pool" on page 220
- "Reducing the number of partially full tapes" on page 220
- "Protecting tapes" on page 221
- "Communicating with the tape management system" on page 225
- "Managing tapes with DFSMShsm installation exits" on page 227

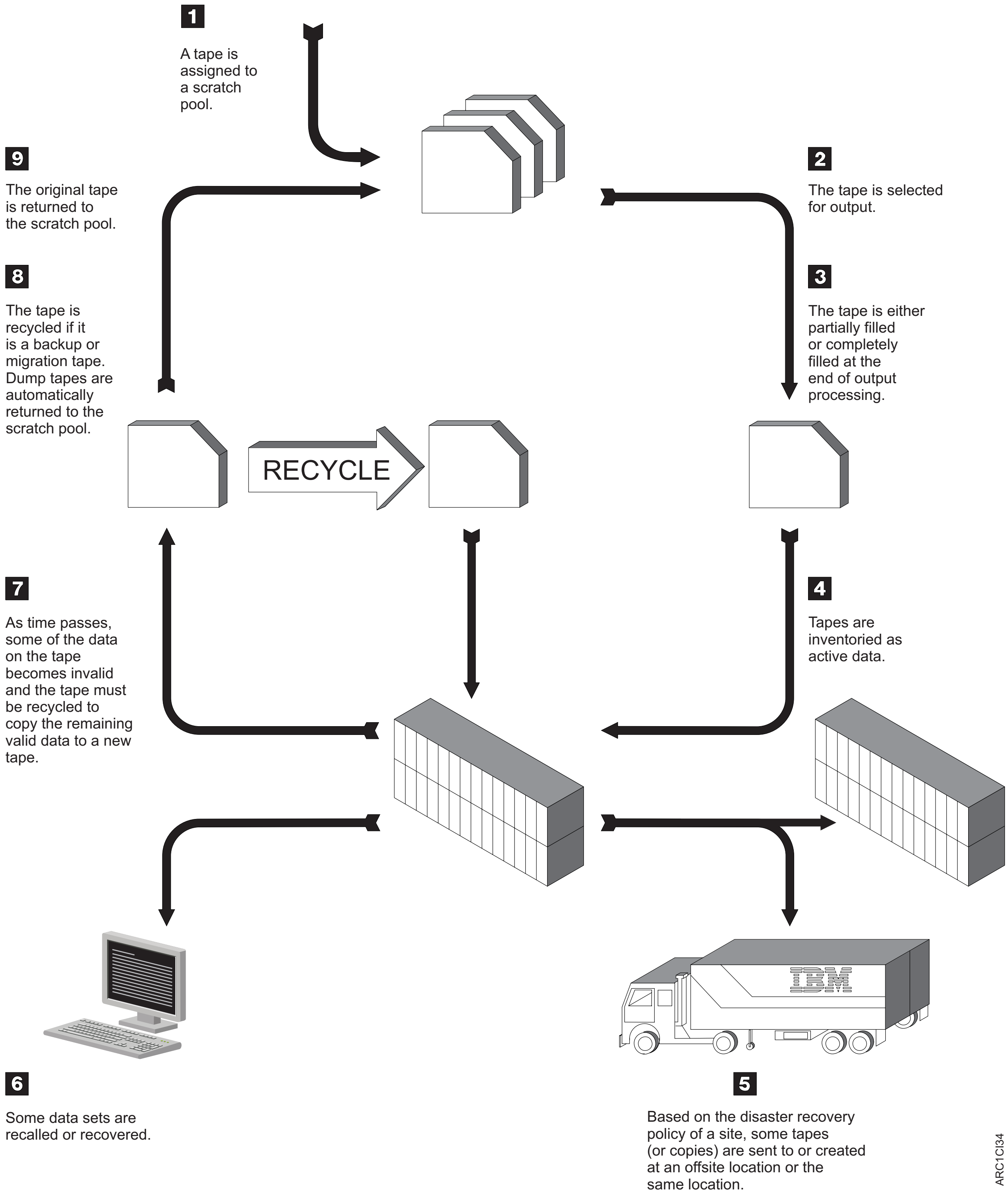


Figure 64. Life Cycle of a Tape. The life cycle of a tape is determined by the SETSYS commands you specify that manage the tape as it enters a scratch pool, is selected for output, is inventoried as active data, is recycled (if it is a backup or migration tape), and finally is returned to the scratch pool.

DFSMShsm tape media

Attention: Reel-type tapes associated with devices prior to 3480 are no longer written for backup and migration functions. However, they are still supported for recall and recover functions.

DFSMShsm tape processing functions support single file tapes. You cannot create either multiple file reel-type backup or migration tapes. Support of reel-type tapes is limited to the following functions:

- Recall or recovery of data sets that currently reside on reel-type tapes
- Creation of dump tapes
- Creation of CDS Version Backup tapes
- ABARS functions

DFSMShsm requires standard labels, but does not support user labels (UHL1 through UHL8), on each type of tape.

Single file cartridge-type tapes

Cartridge-type tapes, associated with 3480 and later tape devices, are always written in single file format, except when they are used as dump tapes. Single file format provides performance advantages for migration and backup processing because it reduces I/O and system serialization. Additionally, single file format provides better recovery for tapes that are partially overwritten or that have become unreadable, because the AUDIT MEDIACONTROLS and TAPECOPY commands work only with single file format tapes.

Single file format reduces I/O and system serialization, because only one label is required for each connected set (as opposed to multiple file format tapes that require a label for each data set). The standard-label tape data set that is associated with the connected set can span up to the allocation limit of 255 tapes. It is possible that HSM could extend the connected set beyond 255 tapes via subsequent backup or migration processing if the last volume was not marked full. This standard-label tape data set is called the DFSMShsm tape data set. Each user data set is written, in 16K logical blocks, to the DFSMShsm tape data set. A single user data set can span up to 254 tapes.

After DFSMShsm writes a user data set to tape, it checks the volume count for the DFSMShsm tape data set. If the volume count is greater than 240, the DFSMShsm tape data set is closed, and the currently mounted tape is marked full and is deallocated. DFSMShsm selects another tape, and then starts a different DFSMShsm tape data set. Data set spanning can be reduced using the SETSYS TAPESPANSIZE command.

Single file format tapes support full DFSMShsm function because they support the tape-copy, tape-replace, and duplex functions of DFSMShsm (multiple file format tapes do not).

Multiple file reel-type tapes

Multiple file format requires a unique standard-label data set for each user data set. Each tape data set has a file sequence number associated with it that can have a value from 1 to 9999. After DFSMShsm writes a user data set to tape, it checks the file sequence number of the file just written. If the sequence number is 9999, the currently mounted tape is marked full and is deallocated. DFSMShsm selects another tape and writes the next user data set with a file sequence number that is one greater than the last file on the newly mounted tape. Empty tapes start with a file sequence number of one.

Obtaining empty tapes from scratch pools

When a DFSMShsm output function fills a tape, it requests another tape to continue output processing. Tapes are obtained from a scratch pool. The scratch pool can be either a global scratch pool or a *specific* scratch pool.

Global scratch pools

A global scratch pool is a repository of empty tapes for use by anyone. The tape volumes are not individually known by DFSMShsm while they are members of the scratch pool. When a scratch tape is mounted and written to by DFSMShsm, it becomes a private tape and is removed from the scratch pool. When tapes used by DFSMShsm no longer contain valid data, they are returned to the global scratch pool for use by anyone and DFSMShsm removes all knowledge of the existence of them.

Global scratch pools are recommended because mount requests can be responded to more quickly and more than when tapes reside in a specific scratch pool. Using a global scratch pool enables easy exploitation of cartridge loaders (including cartridge loaders in tape-library-resident devices) and work well with tape management systems such as DFSMSrmm. Ensure that tapes entered into an SMS-managed tape library global scratch pool are assigned a scratch status.

It is important that global scratch pools be used when multiple tape libraries are assigned to the same storage group. In this scenario, the tape device is selected first, followed by a tape. The tape and device must be in the same library, so using a specific (that is, HSM) scratch pool can result in running out of empty tapes for the tape device that was allocated while empty tapes exists for other tape libraries in the storage group.

Duplex processing always requests a scratch tape whenever it creates an alternate tape. Duplex processing of the alternate tape never uses the specific tape pool. These tapes are returned to the global scratch pool when they no longer contain valid data.

For more information about the role of the scratch pool in tape processing, see Figure 65 on page 209. For more information about the various scratch pool environments, see "Selecting a scratch pool environment" on page 211.

Specific scratch pools

A specific scratch pool is a repository of empty tapes restricted for use by a specific user or set of users. When DFSMShsm is in a specific scratch pool environment, each empty tape as well as each used tape is known to DFSMShsm as a result of being added to the scratch pool, generally by the ADDVOL command. These tapes can be used by only DFSMShsm. The key ingredient of a specific scratch pool is that when an DFSMShsm tape becomes void of data, it is not returned to the global scratch pool but it is retained by DFSMShsm in the specific scratch pool for reuse by DFSMShsm. (You can think of "HSMTAPE" as identifying the pool of specific scratch tapes managed by DFSMShsm.)

Specific scratch pools are not recommended; they are used generally where tapes are owned by individual groups and hence restricted to that group's use, or where the customer's tape management product cannot manage DFSMShsm's tape usage otherwise. These restrictions cause the existence of several mutually exclusive sets of tape and hence increase the size of the overall tape pool as well as the complexity of handling them. When tapes are entered into specific scratch pools of an SMS-managed tape library, ensure that they are assigned a PRIVATE status and that they are associated with the appropriate storage group.

To use specific scratch pools with an SMS-managed VTS (TS7700) library with emulated D/T3490 tape, you must define an esoteric for that library so that DFSMShsm can distinguish between stand-alone drives and emulated drives when processing an ADDVOL command. Use the esoteric with the SETSYS USERUNITTABLE and ADDVOL commands to associate a given tape to the SMS-managed VTS (TS7700) library and DFSMShsm.

You can determine which tapes are in a specific scratch pool (these are your site's specific scratch tapes) by issuing the LIST TTOC SELECT(EMPTY) command as discussed in "Looking at your tape library with the LIST command" on page 198 and in the *z/OS DFSMShsm Storage Administration*.

For more information about the role of scratch pools in tape processing, see Figure 65 on page 209. For more information about the various scratch pool environments, see "Selecting a scratch pool environment" on page 211.

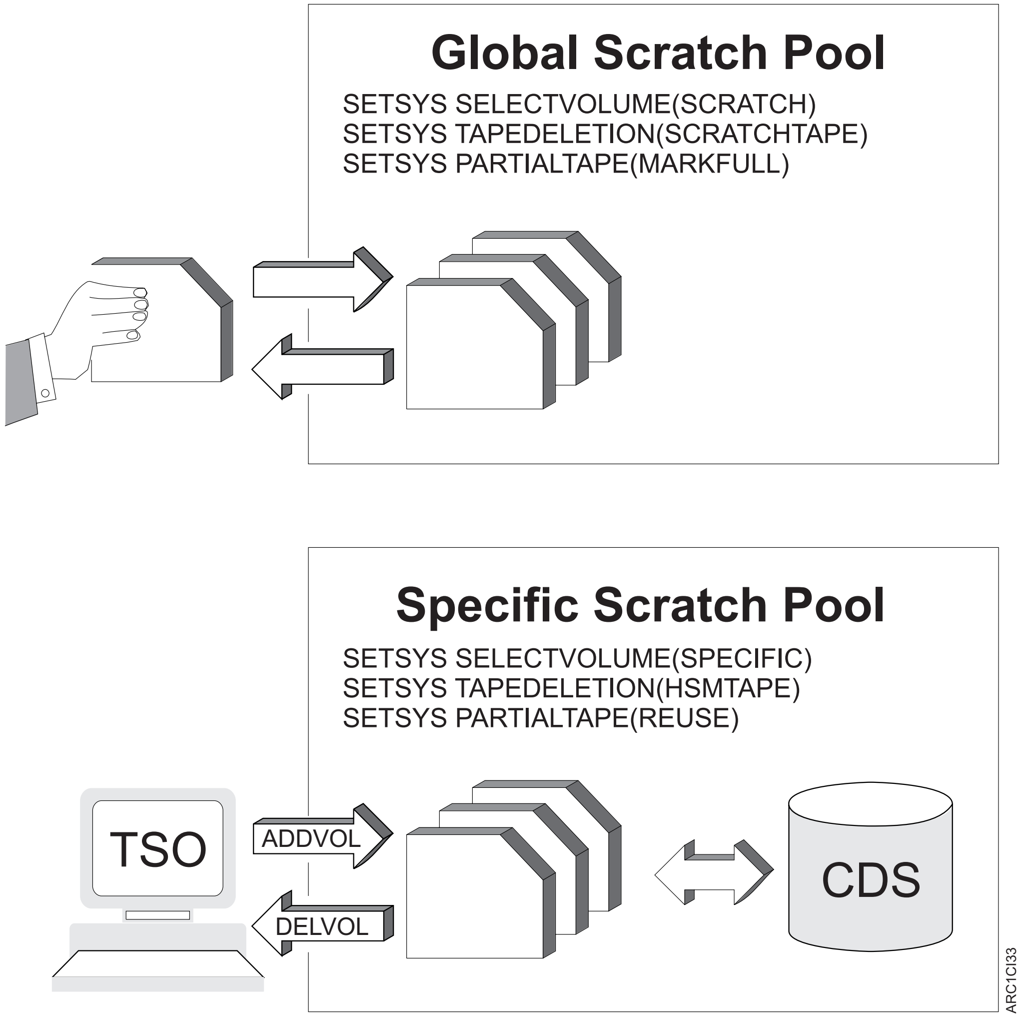


Figure 65. Overview of Tape Scratch Pools. A global scratch pool does not require making tapes known to DFSMShsm because they are managed by a tape management product (DFSMSrmm, for example). Global scratch pools are recommended for SMS-managed tape libraries and unattended operation.

Because DFSMShsm records specific scratch pool activity in the control data sets, a specific scratch pool requires that the operator make tapes known to DFSMShsm by adding tapes to the scratch pool with the DFSMShsm ADDVOL command and by deleting tapes with the DFSMShsm DELVOL command. Specific scratch pools are recommended for nonlibrary environments where specific tapes must be kept physically separate from each other.

Selecting output tape

Because input tape processing always requires a specific tape and involves less configuration than output tape processing, which requires selection of a tape, the majority of tape processing considerations will likely be to output tapes.

Tape hardware emulation

Emulation of one tape subsystem by another has become commonplace. Virtual tape, as well as new technologies, emulates prior technologies. Since the cartridges are not interchangeable, DFSMShsm has added support for several subcategories that indicate that they use 3490 cartridges. They include:

- 3590 Model E 256 track recording
- 3590 Model B 128 track recording

- 3590 Model H 384 track recording
- Virtual Tape Server
- and all others whose device type is 3490

Additionally, DFSMShsm supports the following subcategories of 3590 cartridges:
- 3590 Model E 256 track recording
- 3590 Model B 128 track recording
- 3590 Model H 384 track recording
- and all others whose device type is 3590

This additional support enables multiple use of these technologies concurrently for the same function within the same HSMplex as long as their output use is on different DFSMShsm hosts. This support improves the install of new technologies to improve performance and capacity. When multiple technologies are used concurrently for output outside the DFSMS tape umbrella, an esoteric unit name is needed to enable DFSMShsm to distinguish them. An esoteric name is also needed to select existing (partially filled) tapes that are compatible with desired allocated drives.

Note:
1. The 3590 -E1*x* , -H1*x*, and the 3592-J1A are always in "emulation mode," either as a 3490-E1*x* or 3590-B1*x*. The 3592-E05 and newer tape units only emulate the 3590-B1*x*.
2. You can use the 3592-E05 and newer tape drives only in 3590 emulation mode; never 3490. The 3592 Model J1A can operate in 3490 emulation mode only when using MEDIA5 for output.
3. DFSMShsm allows you to specify 3590-1 as a unit name, provided that all devices that are associated with that name use the same recording technology. In environments where tape device allocations are controlled by other means (such as IBM tape library) consistency checking of the 3590-1 generic unit name can be disabled if necessary. For the details of a patch that you can use, see “Allowing DFSMShsm to use the 3590-1 generic unit when it contains mixed track technology drives” on page 365.
4. To change the volume size used by the IBM TotalStorage Virtual Tape Server, you first need to mark full any partially full virtual migration and backup tapes. See *IBM TotalStorage Virtual Tape Server Planning, Implementing, and Monitoring* section *Migration to Larger Logical Volume Sizes*for implementation details.

Initial tape selection for migration and backup tapes

Initial tape selection is the activity of selecting the first tape for a tape processing function and is always associated with a tape device type. Tapes and devices are selected and allocated differently in an SMS-managed tape library environment than in a non-SMS-managed, nonlibrary environment.

DFSMShsm selects tapes in the following order:
1. Partial (only if eligible and available). If you are selecting a migration tape, DFSMShsm begins by considering the tape last used by the task. If it is not acceptable, then it selects the eligible partial tape with the highest percentage of written data.
2. Empty
3. Scratch

In a nonlibrary environment, DFSMShsm selects a tape that is compatible with the unit type restriction for the function.

In an SMS-managed tape library, MVS allocates a device according to the data class, storage class, and storage group associated with the data set. DFSMShsm, using the above selection order, selects a tape that is compatible with that device.

Partial tapes selected in a duplex environment must have a duplex alternate. (A nonduplexed alternate is an alternate created by user-initiated TAPECOPY.) Empty or scratch tapes selected in a duplex environment will have a scratch tape for the alternate.

Subsequent tape selection for migration and backup tapes

Subsequent tape selection, controlled by the SETSYS SELECTVOLUME command, takes place when a tape currently being written reaches end-of-volume and another tape is required to continue processing. Subsequent tape selection relates to demounting a tape and mounting another tape on a tape device that is already allocated. The alternate tape of a duplex pair will always be a scratch tape.

Initial and subsequent selection of dump tapes

Because dump processing always selects an empty tape, initial and subsequent tape selection are the same and they are controlled by the SETSYS SELECTVOLUME command. The order of tape selection and device selection for dump processing is the same as the order for migration and backup processing discussed in "Initial tape selection for migration and backup tapes" on page 210.

Selecting a scratch pool environment

The scratch-pool environment is defined by three SETSYS parameters: SELECTVOLUME, PARTIALTAPE, and TAPEDELETION.

The SELECTVOLUME parameter controls whether subsequent tape mounts for migration processing and backup processing are specific or nonspecific, and controls all tape mounts for dump processing.

The PARTIALTAPE parameter controls whether to mark migration and backup tapes full (so a different tape is requested the next time the function runs) after processing, and whether initial selection for migration processing and backup processing selects a partially written tape (from the last time the function ran) or a scratch tape.

The TAPEDELETION parameter controls whether emptied tapes are returned to a global scratch pool or retained in the (DFSMShsm-managed) specific scratch pool, and whether the scratch pool is global or specific.

Note: The three SETSYS commands listed above affect only original tape selection. Duplex processing always requests a scratch tape whenever it creates an alternate tape. It never uses the specific tape pool.

For a detailed discussion of these commands, see *z/OS DFSMShsm Storage Administration*; however, for your convenience four of the most common tape environments are graphically described in this section. By issuing the SETSYS commands in our examples, you take a fast path to defining your scratch-pool environment.

- Figure 66 on page 212 supports the most automation and is the easiest to manage

- Figure 67 on page 213 is the most efficient in utilizing tape media
- Figure 68 on page 214 supports separate management of ranges of tape serial numbers while exploiting ACL automation
- Figure 69 on page 215 is an environment for sites that want only specific scratch mounts and do not want to use a tape management program

Performance tape environment with global scratch pool for library and nonlibrary environments

Figure 66 shows a tape environment that is recommended for sites that want to and quickly implement tape processing for either library or nonlibrary tape environments.

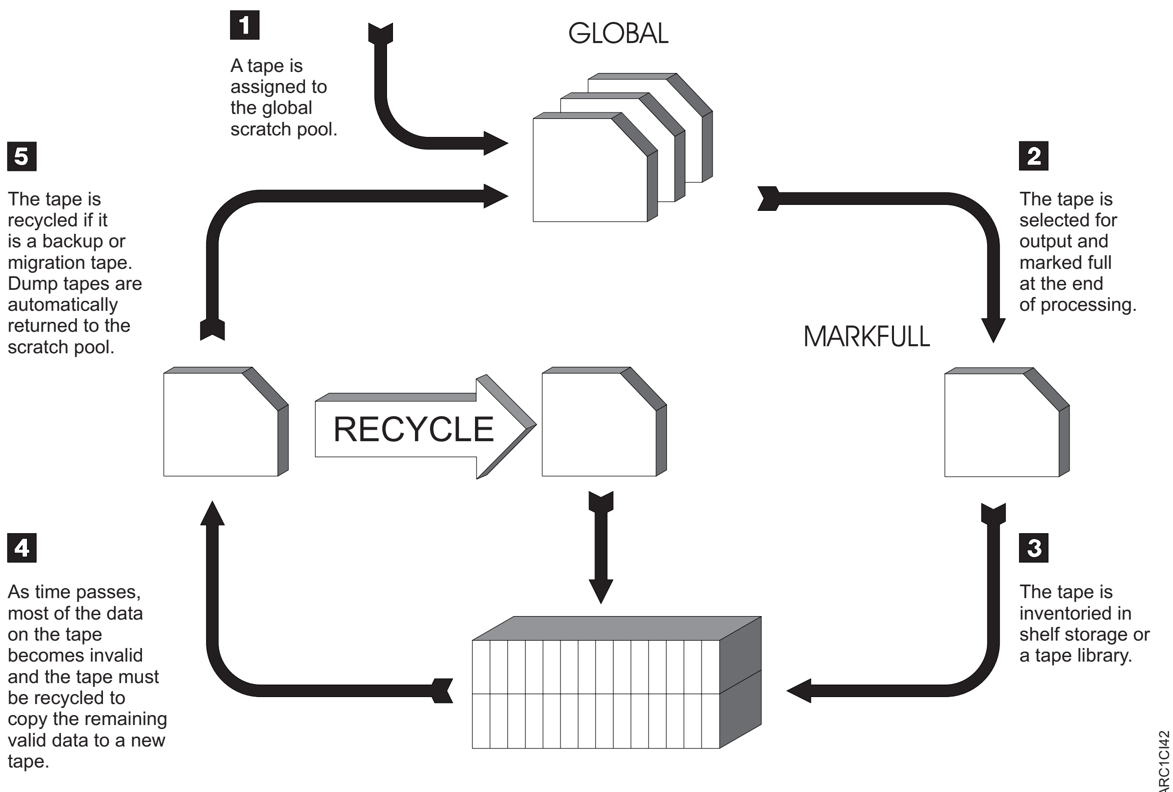


Figure 66. Recommended Performance Tape Processing Environment. This is the most versatile and easily managed of the recommended tape environments. This environment is recommended for both library and nonlibrary tape environments. Note that DFSMShsm treats a partial migration tape that is taken away for recall as if you had specified SETSYS PARTIALTAPE(REUSE).

Table 21 on page 213 summarizes the performance characteristics of the global scratch pool environment.

Table 21. Summary: Performance Global Scratch Pool Environment

Tape Environment Definition	Features	Trade-offs
SETSYS - SELECTVOLUME(SCRATCH) - TAPEDELETION(SCRATCHTAPE) - PARTIALTAPE(MARKFULL)	• Optimizes tape libraries because the robot can fill cartridge loaders with nonspecific scratch tapes when it is not processing tapes • Optimizes cartridge-loader exploitation in either a library or a nonlibrary environment • Easier to manage than specific scratch pools.	• Does not fully utilize tape capacity • Requires more recycle time than the tape-capacity optimization environment

Tape-capacity optimization tape environment with global scratch pool for library and nonlibrary environments

Figure 67 shows a tape environment that is recommended for sites that want to optimize utilization of tape cartridges.

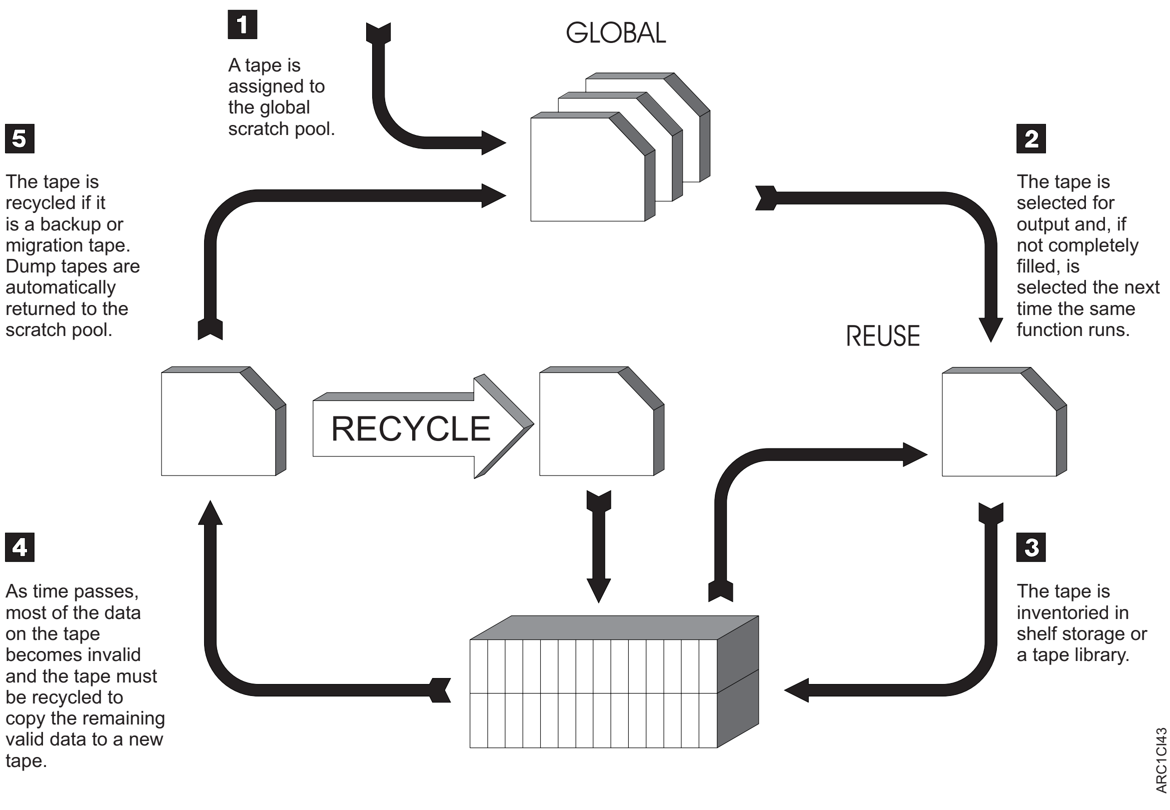


Figure 67. Recommended Tape Processing Environment that Maximizes Tape Capacity. This tape environment differs from the performance tape environment only in the way partially filled tapes are managed. Because partially filled tapes are selected the next time the same function runs, the tape's media is fully utilized.

Table 22 on page 214 is a summary of tape-capacity optimization considerations in a global scratch pool environment.

Table 22. Summary: Tape-Capacity Optimization Global Scratch Pool Environment

Tape Environment Definition	Features	Trade-offs
SETSYS - SELECTVOLUME(SCRATCH) - TAPEDELETION(SCRATCHTAPE) - PARTIALTAPE(REUSE)	• Utilizes full tape capacity for both original and alternate tapes • Effective for tape libraries • Requires less recycle time than the performance tape environment because fewer tapes need to be recycled	• Does not fully exploit cartridge loaders for the first mount of each task, of each function, each day. • TAPECOPY processing does not copy the final tape of each task until that tape is filled.

Media optimization for DFSMShsm-managed nonlibrary tape environment

Figure 68 shows an environment that supports separated ranges of tape-volume serial numbers (for example, the range of tapes managed by DFSMShsm is separate from ranges of tapes used by other departments for other purposes).

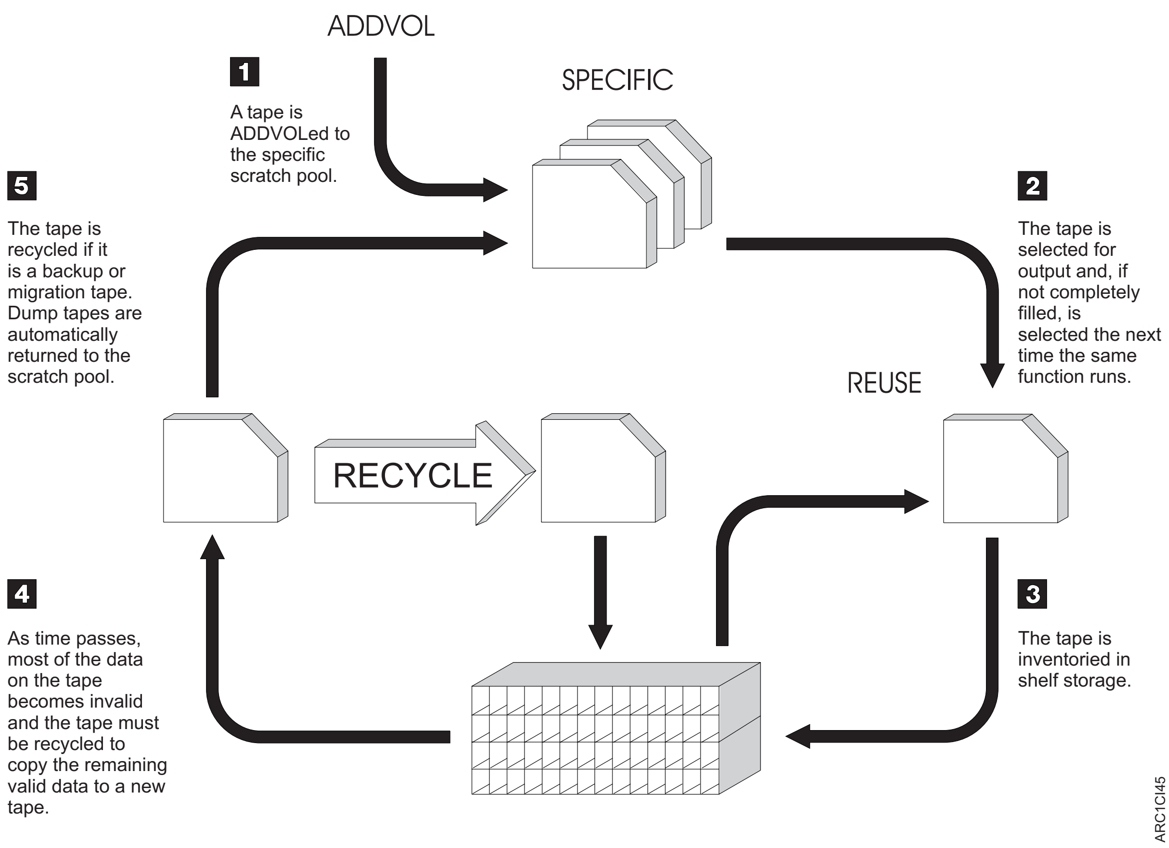


Figure 68. Recommended Environment For Managing Separate Ranges of Tapes. In this environment, operations requests a list of empty tapes. The initial mount request for migration, backup, or their recycle is for a specific tape. After the initial tape is mounted, operations either preloads the empty tapes into cartridge loaders or responds to mount requests.

Table 23 on page 215 is a summary of considerations for a DFSMShsm-managed tape specific scratch pool environment for media optimization.

Table 23. DFSMShsm-Managed Tape Specific Scratch Pool Environment for Media Optimization

Tape Environment Definition	Features	Trade-offs
SETSYS - SELECTVOLUME(SCRATCH) - TAPEDELETION(HSMTAPE) - PARTIALTAPE(REUSE)	• Optimizes tape utilization • Effective for partially unattended operation (you must manually mount the initial tape). You can preload ACLs for subsequent tape selection. • Separate management of ranges of tape serial numbers • Does not require a tape management system	• Requires more management than a global scratch pool • Empty tapes must be DELVOLed to return them to scratch status

Performance optimization for DFSMShsm-managed nonlibrary tape environment

Figure 69 shows an environment for sites that want only specific scratch mounts; DFSMShsm manages the tapes and no tape management program is required.

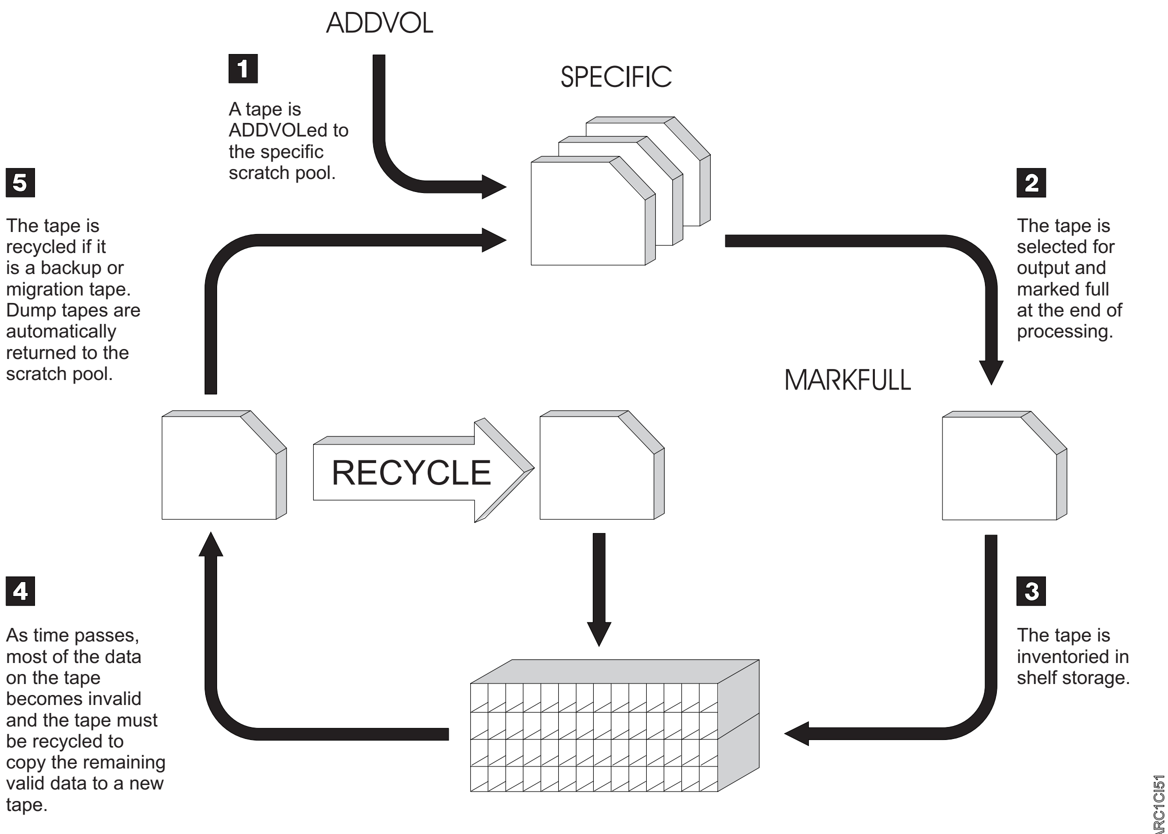


Figure 69. Environment For Specific Scratch Mounts. In this environment, DFSMShsm manages tapes and requests specific tapes to be mounted. An operator is required for all tape mounts; however, separate ranges of tapes can be assigned to DFSMShsm and to other users and managed according to the requirements of each user. Note that DFSMShsm treats a partial migration tape that is taken away for recall as if you had specified SETSYS PARTIALTAPE(REUSE).

Table 24 on page 216 is a summary of considerations for a DFSMShsm-managed tape specific scratch pool environment for performance optimization.

Table 24. DFSMShsm-Managed Tape Specific Scratch Pool Environment for Performance Optimization

Tape Environment Definition	Features	Trade-offs
SETSYS - SELECTVOLUME(SPECIFIC) - TAPEDELETION(HSMTAPE) - PARTIALTAPE(MARKFULL)	• Separate management of ranges of tape serial numbers • Does not require a tape management system. DFSMShsm manages tapes and requests specific tape mounts.	• Requires more management than a global scratch pool • Tapes must be initially ADDVOLed and eventually DELVOLed

Implementing a recycle schedule for backup and migration tapes

Recycle is the activity of moving valid data from old DFSMShsm tapes to new tapes. This becomes necessary for the following reasons:

- To produce scratch tapes. A migrated data set is considered to be invalid when it is deleted, expired, or individually recalled. A backup version of a data set is invalid when it is deleted, expired, or has become excess and is rolled-off.
- To move data to new tape technology or tape media.
- To refresh data recorded on tapes. Data written to tape media is considered readable for about 10 years.

Recycle processing separates valid and invalid data sets and consolidates valid data sets on fewer tapes, therefore making more scratch tapes available for reuse. Figure 70, Figure 71 on page 217, and Figure 72 on page 217 illustrate the concept of recycle processing:

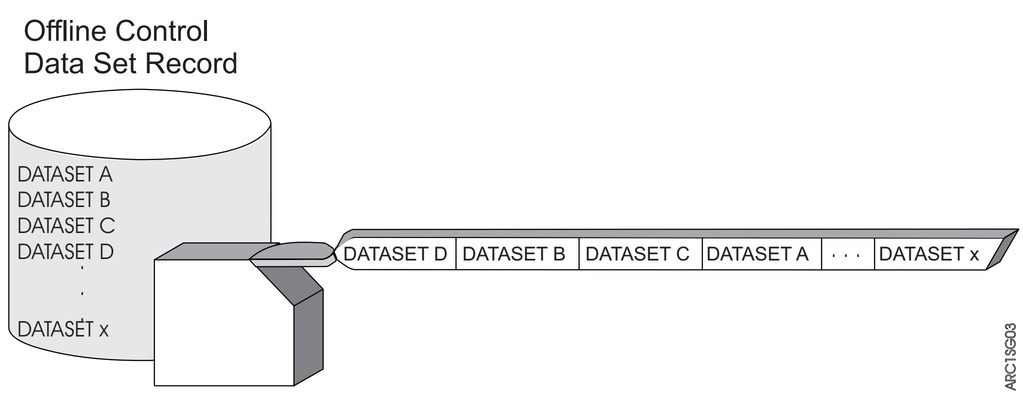


Figure 70. Creation of Tape Data Set Copies. DFSMShsm makes an entry in the offline control data set record for each data set stored on a migration or backup tape.

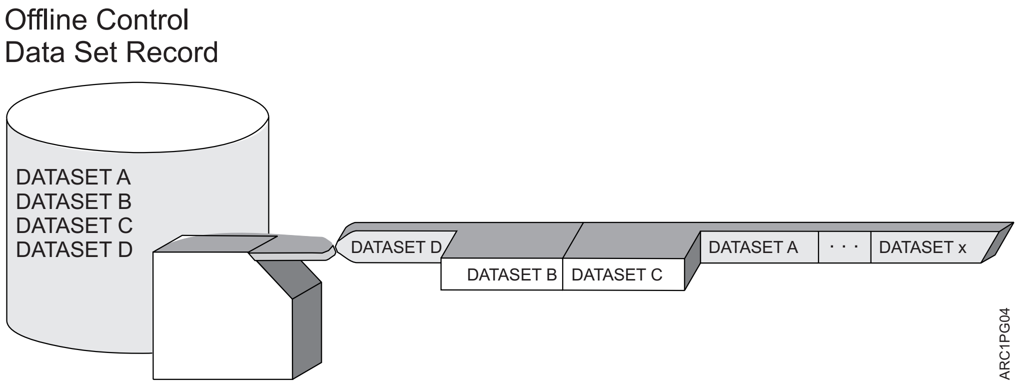


Figure 71. Tape Data Set Copy Invalidation. As migrated data sets are recalled and as backup versions expire, the data set entries in the OCDS become invalid. Invalid data sets remain on the tape, however, until the tape is recycled.

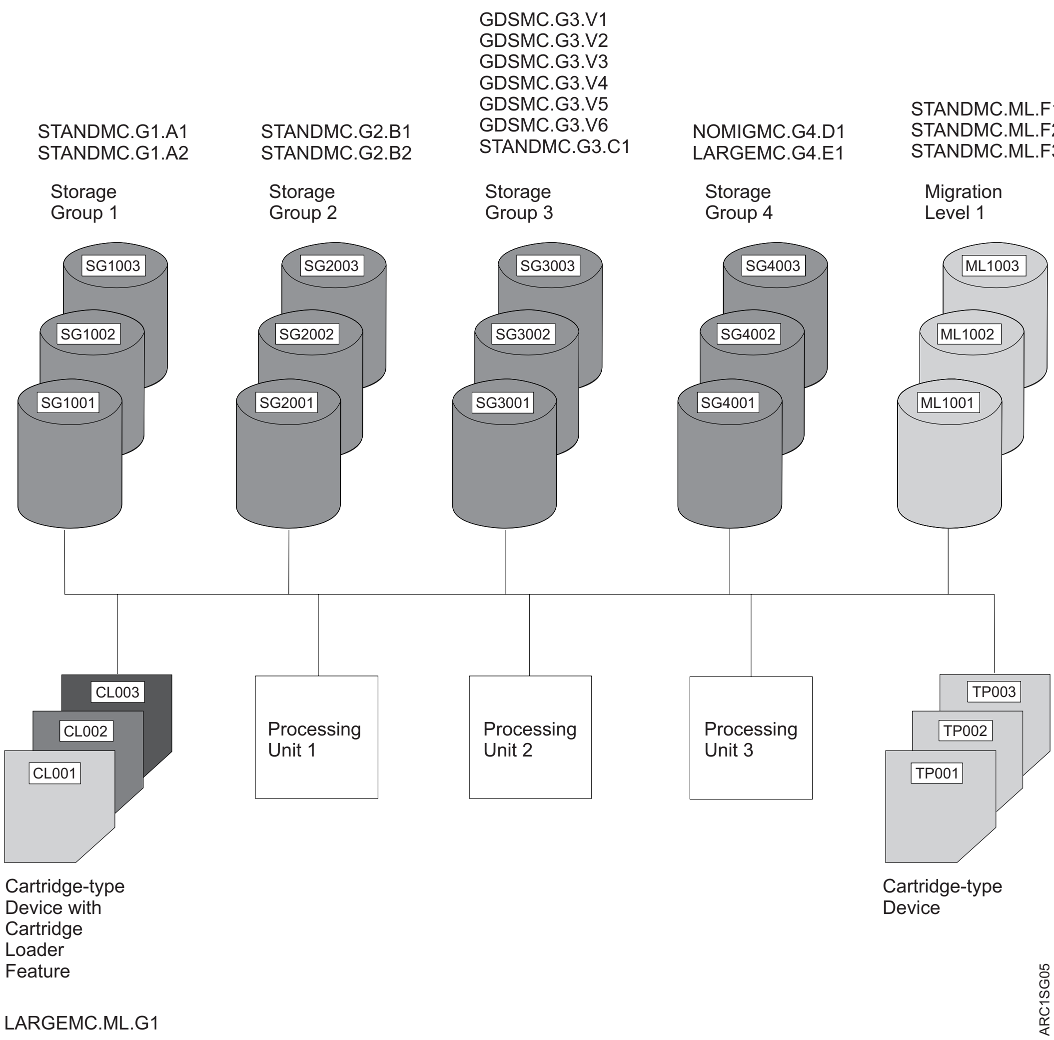


Figure 72. Tape Efficiency Through Recycle Processing. As a result of recycle processing, valid data sets from source tape 1 are copied to the recycle target tape. After all valid data sets from source tape 1 are copied, valid data sets from source tape 2 are copied to any recycle target tapes. Source tapes that have been recycled are returned to the scratch pool.

You can choose to run recycle on a regular schedule or you can choose to wait until DFSMShsm uses enough tapes that you need to make more tapes available.

Because recycle can be run independently for migration tapes and for backup tapes, the schedules can be independent of each other. Because recycle is operating only on DFSMShsm-owned volumes, you can schedule it whenever your tape devices are available.

When to initiate recycle processing

When planning to implement a recycle schedule, consider the number of scratch tapes that you need, the number of recycle tasks that you want to run, tape drive contention, and where you want to see messages generated during processing.

Number of scratch tapes needed: By using the RECYCLE LIMIT parameter, you can specify the net (emptied minus used) number of tapes to be returned to scratch status. Recycle processing is quiesced when the limit is reached. Because each recycle task completes the connected set currently being processed, a few tapes over the specified limit are expected to be freed.

Number of recycle tasks to run: Up to 15 recycle tape processing tasks can run simultaneously in a host, with two tape drives required for each task, one for input and one for output. (Duplex processing requires three tapes drives for each task, one for input and two for output.) By using the SETSYS MAXRECYCLETASKS parameter, the number of recycle tape processing tasks can be changed dynamically, even during recycle processing, to any number from one to 15. The default for this parameter is two recycle processing tasks.

Tape drive contention: In a tape environment where contention for tape drives may be a consideration, Table 25 shows BACKUP(*bfreq*) or MIGRATION(*mfreq*) values recommended when recycling single-file-format cartridges. BACKUP(*bfreq*) and MIGRATION(*mfreq*) are subparameters of the SETSYS RECYCLEINPUTDEALLOCFREQUENCY command.

Table 25. Recommended RECYCLEINPUTDEALLOCFREQUENCY Values

Type of Environment	Value	Resulting Allocation
A single IBM 3494/3495 Tape Library Dataserver environment having all compatible drives	0	The input unit is allocated for the duration of recycle processing.
An SMS-managed tape environment that has multiple IBM 3494/3495 machines where inputs can come from different ATLs	0	A unique allocation is allowed for each input connected set.
A BTLS-managed environment that has multiple IBM 3494/3495 machines where inputs can come from different ATLs	1	RECYCLE allocates tape drives for each connected set in the correct library.
A multiple non-SMS-managed tape library where not all backup or migration tapes are in a single library (for example, an STK ACS cluster where you want to prevent pass-through processing)	1	RECYCLE allocates tape drives for each connected set in the correct library.
An environment that has incompatible tapes even though they may appear the same, including when one has a mix of real and emulated devices, such as 3490s and emulators of 3490s	1	RECYCLE allocates tape drives for each connected set.
A tape environment with manual tape operators, but no incompatible mix of real and emulated devices appearing as the same device	5 or 10	New allocations are allowed every 5 to 10 input connected sets.

Note: Input units are always deallocated between connected sets when the tape volumes are not single file format cartridges.

Generated messages: You can send messages generated during recycle processing or a list of tapes specified with the DISPLAY parameter to the output data set by specifying the RECYCLE OUTDATASET *dsname* parameter. The list of tapes is in volume-serial sequence (most likely not in tape-mount sequence). When the number of volumes is not too large, operators can use this list to pull tapes from shelves.

However, if you know or suspect that you have a large number of tapes to pull manually from tape storage, there are other options to use. For example, the VERIFY and EXECUTE TAPELIST parameters allow you to divide the total number of volumes to be recycled for one category into smaller subsets (pull groups), each in volume-serial sequence. You can then send these subsets to the output data set having either a name with a prefix specified with the TAPELIST(PREFIX) parameter or a fully qualified name specified with the TAPELIST(FULLDSNAME) parameter.

How long to run recycle processing

Another consideration when planning to implement a recycle schedule is how long you want recycle processing to run. Consider the number of required scratch tapes, the maximum percentage of tape media you want used, tape drive contention, and recycle performance.

Number of required scratch tapes: By using the RECYCLE LIMIT parameter, you can specify the net number of tapes to be returned to scratch status. Recycle processing is quiesced when the limit is reached. Because each recycle task completes the connected set currently being processed, a few tapes over the specified limit are expected to be freed.

Maximum percentage of valid data: With the RECYCLE PERCENTVALID parameter, you can specify a maximum percentage of tape media that is occupied by valid data for each functional category of DFSMShsm-owned tape and be a candidate for recycle. You can also tell DFSMShsm the functional category or all categories to be processed. DFSMShsm orders the list and selects for processing the ones with the least amount of valid data and stops if the limit is reached. The valid data is read from the selected tapes and written to selected output tapes, filling the output tapes with valid data and reducing the number of DFSMShsm-owned tapes.

Tape drive contention: When determining how long to run recycle processing, you should consider how much contention you have for tape devices. With the SETSYS RECYCLEINPUTDEALLOCFREQUENCY parameter, you can cause DFSMShsm to periodically deallocate an input unit during recycle processing. For a list of recommended values that you can use when you are recycling single-file-format cartridges, see Table 25 on page 218.

Recycle performance: A host can run only one RECYCLE command at a time. One RECYCLE command, however, can run up to 15 recycling tasks simultaneously. Two tape drives are required for each task, one for input and one for output. (Duplex processing requires three tapes drives for each task, one for input and two for output.) The more tape drives you use, the faster recycle processing proceeds.

For example, if you issue a recycle command against a specific tape volume, processing is equivalent to one recycle task, requiring two (or three with duplex processing) tape drives. If you issue a recycle command against all ML2 tape volumes and SETSYS MAXRECYCLETASKS is set to 15, DFSMShsm processes 15 tapes at the same time, requiring 30 (or 45 with duplex processing) tape drives.

The SETSYS MAXRECYCLETASKS parameter allows you to dynamically change the number of recycle tape processing tasks, even during recycle processing, to any number from one to 15. If you decrease the tasking level below what is currently running, tasks are quiesced until the new level is reached. Currently running tasks are not stopped in the middle of the volume set they are processing. The default is two recycle-processing tasks.

Returning empty tapes to the scratch pool

If DFSMShsm considers tapes to be empty, they can be returned to the scratch pool in a number of ways, depending on the function that generated the data on the tape. Table 26 summarizes the various ways empty tapes are returned to the scratch pool.

Table 26. How Different Functions Return Empty Tapes to the Scratch Pool

Function	Recycle Method
Backup/Migration	Backup and migration tapes are returned to scratch status as a result of being recycled with the RECYCLE command. Even if they contain no valid data, migration and backup tapes must be recycled to verify that residual data is invalid. Alternate tapes are returned along with their originals.
Dump	Dump tapes can be returned to scratch status as soon as they expire during automatic dump processing. An empty dump tape is managed according to the specifications in the dump class associated with that tape.
Control data set backup	Control data set backup tapes are returned to the scratch pool when their resident backup-version data sets roll off.
ABARS	ABARS tapes are returned to the scratch pool when their resident backup-version data sets roll off or expire.

For any of the functions in Table 26, DFSMShsm invokes the EDGTVEXT exit to communicate the status of empty tapes to DFSMSrmm, and it only invokes the tape volume exit (ARCTVEXT) if SETSYS EXITON(TV) has been specified. For more information about DFSMShsm installation exits, refer to *z/OS DFSMS Installation Exits*.

Reducing the number of partially full tapes

When an ML2 tape that is currently being used as a migration or recycle target is needed as input for a recall or ABACKUP, the requesting function can take the tape away whenever migration or recycle completes processing its current data set. When recall or ABACKUP is finished with such a partial tape, DFSMShsm leaves it available as a migration or recycle target again, regardless of the PARTIALTAPE setting.

During initial selection for an ML2 tape for output from a migration or recycle task, DFSMShsm tries to select the partial tape with the highest percentage of written data. An empty tape is selected only if no partial tapes are available.

There are three related options for the LIST TTOC command:

- SELECT(RECALLTAKEAWAY) lists only those ML2 tapes taken away by recall. (If this list is too extensive, you may want to tune your ML2 tape migration criteria.)
- SELECT(ASSOCIATED) lists only those ML2 partial tapes that are currently associated with migration or recycle tasks as targets on this host or any other in the HSMplex (two or more hosts running DFSMShsm that share a common MCDS, OCDS, BCDS, and journal).
- SELECT(NOTASSOCIATED) lists only those ML2 partial tapes that are not associated with migration or recycle tasks as targets.

The RECYCLE command for ML2 can recycle some of these partial tapes. By using the SETSYS ML2PARTIALSNOTASSOCIATEDGOAL parameter, you can control the tradeoff between having a few under-used ML2 tapes versus the time needed to recycle them. If the number of nonassociated ML2 partial tapes in the HSMplex that is meeting the PERCENTVALID and SELECT criteria for recycle exceeds the number specified by the SETSYS parameter, recycle includes enough partial tapes to reduce the number of partial tapes to the specified number. Recycle selects those tapes to process in the order of least amount of valid data to most amount of valid data.

Protecting tapes

Protecting DFSMShsm's tapes allows only DFSMShsm to access its tapes. Users can obtain data sets from DFSMShsm-owned tapes through DFSMShsm, but they cannot directly allocate or read the tapes.

The SETSYS TAPESECURITY command sub-parameters determine how DFSMShsm tapes are protected. The tape security options follow:

- RACF protection through RACF DATASET class with
  - (EXPIRATION and EXPIRATIONINCLUDE) or
  - (PASSWORD)
- RACF protection through RACF TAPEVOL class through
  - (RACF and RACFINCLUDE)
- Expiration-date protection through
  - (EXPIRATION and EXPIRATIONINCLUDE)
- Password protection through
  - (PASSWORD) - This is the default, if no selection is made.

As you implement your tape-protection policies, you must consider whether your tapes are managed by DFSMShsm alone, or if a tape management system (DFSMSrmm, for example) co-manages tapes. If a tape management system shares tape management responsibility, tape protection requirements are different than if DFSMShsm alone is managing tapes.

In some cases, you can specify more than one of the protection options from the preceding list. For example, you can choose the RACF and EXPIRATION sub-parameters to indicate that you want both RACF and expiration date protection.

For more information about using a tape management system and security, see "Defining the environment for the tape management system" on page 225.

RACF protection

You should be familiar with the Protecting Data on Tape section in Chapter 6, Protecting Data Sets on DASD and Tape of *z/OS Security Server RACF Security Administrator's Guide*, when implementing a RACF solution.

RACF can protect tapes using TAPEVOL profiles, or DATASET profiles, or both. You can direct DFSMShsm to add TAPEVOL protection to tapes it selects for output, and remove that protection automatically when it releases the tapes to the scratch pool. DFSMShsm cannot remove protection if the entire scratch pool is protected by RACF, in which case users cannot allocate or read the tapes directly. As tapes become empty, RACF TAPEVOL protection is removed and the tapes can be reused immediately; whereas tapes with expiration-date and password protection might need to be reinitialized (as determined by your tape management procedures) before a global scratch pool can reuse them. DFSMShsm can protect tape volumes with RACF by adding them to a RACF tape-volume set (HSMHSM, HSMABR or DFHSMx). All tapes in DFSMShsm's RACF tape-volume set share the same access list and auditing controls.

You should use RACF profiles to protect all HSM tapes. You can use RACF TAPEVOL profiles with the SETSYS TAPESECURITY(RACF|RACFINCLUDE) option, or use RACF DATASET profiles. RACF DATASET profiles may be used in conjunction with any of the other tape security options.

Using RACF DATASET class profiles: You can implement tape data set protection for DFSMShsm volumes by using DATASET profiles just as you can for any other DASD or tape data set in the system. This is applicable for installations that do not want to use the SETSYS TAPESECURITY values of either RACF, or RACFINCLUDE, because you do not want the RACF TAPEVOL class to be active. If you do not use RACF or RACFINCLUDE, then you must use EXPIRATION, EXPIRATIONINCLUDE, or the default of PASSWORD.

You have two choices for using RACF DATASET class profiles:
- Driven by SETROPTS TAPEDSN
- Driven by DEVSUPxx TAPEAUTHDSN=YES

In either case you must have generic DATASET class profiles created that cover each of the data set name prefixes that DFSMShsm currently uses:
- **ADDSD 'mprefix.**'**: Specifies the DFSMShsm-defined migrated data set prefix.
- **ADDSD 'bprefix.**'** : Specifies the DFSMShsm-defined backup and dump data set prefix.
- **ADDSD 'authid.**'**: Specifies the DFSMShsm prefix used for control data set backups.
- **ADDSD 'bprefix.DMP.**'**: If you want dump tapes to be protected in a different way to backup tapes.
- **ADDSD 'outputdatasetprefix.**'**: Specifies that ABARS created aggregates are protected.

The access list for these data set profiles should only have the DFSMShsm or ABARS userid added to it. For more information, see "Creating user IDs" on page 170.

For information about security options, see *z/OS Security Server RACF Security Administrator's Guide*.

For information about the DEVSUPxx parmlib member, see *z/OS MVS Initialization and Tuning Reference*.

For more information about protecting DFSMShsm tape volumes with TAPEVOL, see "Protecting tapes" on page 221.

Conversion from RACF TAPEVOL to RACF DATASET class profiles

Determine whether you will use TAPEDSN in RACF or TAPEAUTHDSN=YES in DEVSUPxx and define the DATASET profiles (refer to "Using RACF DATASET class profiles" on page 222). DFSMShsm maintains volumes in the TAPEVOL sets for HSM and ABARS tapes until you inactivate the TAPEVOL class.

RACF DATASET Class profiles should be used with TAPESECURITY(EXPIRATION, EXPIRATIONINCLUDE or PASSWORD). After changing the TAPESECURITY parameter, any new scratch tapes will be protected using expiration dates or password. Either of these protection cases may require tape initialization before reuse by the global scratch pool. Leaving TAPEVOL class active and HSMHSM, HSMABR and DFHSMx profiles defined ensures that tapes in DFSMShsm's current inventory remain protected by RACF TAPEVOL until returned to scratch. When there are no volumes defined to the TAPEVOL profiles, you can delete the HSMHSM, HSMABR, and DFHSMx tape profiles. If you implement RACF DATASET Class Profiles and deactivate the TAPEVOL class, DFSMShsm will issue an error message when returning tapes protected by RACF TAPEVOL to scratch.

Expiration date protection

Expiration date protection protects tapes by preventing users from overwriting the tapes without operator or tape management system intervention; it does not prevent users from directly allocating and reading the tapes. See "Protecting tapes" on page 221 for a discussion of protecting aggregate tapes.

You can use expiration date protection for environments where DFSMShsm and a tape management system (DFSMSrmm, for example) co-manage tapes.

A more comprehensive solution is to use policies defined with a function like "vital record specification" in a tape management product like DFSMSrmm. For more information about tape management systems, see "Communicating with the tape management system" on page 225.

Defining expiration date protection for tapes: DFSMShsm places a default expiration date of 99365 in the IBM Standard Data Set Label 1 (HDR1, EOV1, and EOF1). Single file format tapes have only one data-set label. You may want to change the expiration date of tapes to something other than the default to communicate to a tape management program that there is something special about these tapes. You can 1) change the expiration date that DFSMShsm places on migration and backup tapes with the ARCTDEXT exit, 2) change the date DFSMShsm places on ABARS tapes with the ARCEDEXT exit, or 3) change a dump tape's expiration date with the TAPEEXPDT parameter of the DEFINE command when defining a dump class. **Note:** Setting a date of 00000 is the equivalent of not having set an expiration date. For more information about the DFSMShsm installation exits, refer to *z/OS DFSMS Installation Exits*.

Note: The EXPIRATION and EXPIRATIONINCLUDE options are equivalent for dump processing, because data sets are not individually processed during volume dump.

Password protection

Password protection protects tapes by setting a password indicator in the standard tape labels. An X'F1' appears in the data set security byte in the IBM Standard Data Set Label 1 (HDR1, EOV1, and EOF1) of each backup, migration, and dump tape. Single file format tapes have only one data set label.

Guideline: IBM does not suggest the use of data set password protection alone, because it provides less protection than the use of RACF. However, it may be used in conjunction with RACF DATASET Class Profiles. See *z/OS DFSMSdfp Advanced Services* for a discussion of password protection.

Dump tape security considerations

Tape security options are the same for dump tapes, backup tapes, and migration tapes; however, there are some differences in implementation because DFSMShsm does not actually write on the dump tapes.

Restrictions for changing the tape-security options apply to dump-tape processing, except that the same security options that were in effect at the start of a dump operation remain in effect for the duration of the dump operation, even though the options can be changed with the SETSYS command at any time. This restriction is because dump processing marks any partially filled tapes as full; they are never selected for additional dump output. See Table 27 for a list of options and SETSYS command attributes.

Dump tapes have no restriction (unlike backup and migration tapes) about placing password-protected data sets on non-password-protected tapes.

Attention: Dump tapes that are password protected by DFSMShsm cannot be restored directly by DFSMSdss. Password-protected dump tapes must be write protected (ring removed or cartridge knob set) before they are mounted for restore processing.

Table 27. DFSMShsm Tape Security Options

Security Option	Commands	Description
RACF	SETSYS TAPESECURITY (RACF)	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 sub-parameter does not support backup or migration of password-protected data sets.
	SETSYS TAPESECURITY (RACFINCLUDE)	DFSMShsm protects each backup, migration, and dump tape and additionally backs up and migrates password-protected data sets to non-password-protected tapes. DFSMShsm also protects migration and backup tapes generated as a result of TAPECOPY processing.
RACF DATASET Class Protection	TAPESECURITY(EXPIRATION | EXPIRATIONINCLUDE) or TAPESECURITY(PASSWORD)	DFSMShsm protects each backup, migration, dump and ABARS tape using a tape data set name profile.
Expiration Date Protection	SETSYS TAPESECURITY (EXPIRATION)	DFSMShsm protects each backup, migration, and dump tape with an expiration date. The EXPIRATION sub-parameter does not support backup or migration of password-protected data sets.

Table 27. DFSMShsm Tape Security Options (continued)

Security Option	Commands	Description
	SETSYS TAPESECURITY (EXPIRATIONINCLUDE)	DFSMShsm protects each backup, migration, and dump tape and additionally backs up and migrates password-protected data sets to non-password-protected tapes.
Password Protection	SETSYS TAPESECURITY (PASSWORD)	DFSMShsm protects each backup, migration, and dump tape with a password. Aggregate backup tapes cannot be password-protected.

Removing the security on tapes returning to the scratch pool

The security types of RACF protection, expiration-date protection, and password protection affect your choice of returning empty tapes to scratch pools. DFSMShsm automatically removes RACF protection for tapes protected by the RACF TAPEVOL class before returning them to a global scratch pool or to a specific scratch pool.

DFSMShsm cannot remove the password and expiration date protection. These tapes may require re-initialization before they can be returned to the global scratch pool.

for more information about ABARS tape processing, see *z/OS DFSMShsm Storage Administration*.

Communicating with the tape management system

Many sites manage tapes with a tape management system (DFSMSrmm, for example) and DFSMShsm. A tape begins its life (see Figure 64 on page 205) as a scratch tape, is used by DFSMShsm to store data, and is returned to the tape management system to be reused as a scratch tape. To implement this form of concurrent tape management, communications must be coordinated whenever you define the environment and data sets for the use of a tape management system.

Although the following problem does not occur with DFSMSrmm, it is possible for some tape management systems to incorrectly record DFSMShsm ownership of a tape mounted during a nonspecific request even though DFSMShsm later rejects the tape; for example, the user mounts a RACF-protected tape for a nonspecific request but DFSMShsm, for whatever reason, rejects the tape. Normally, DFSMShsm only accepts ownership after it performs checks using the DCB Tape Validation Exit; however, some tape management systems record DFSMShsm ownership earlier in the mounting sequence. If DFSMShsm rejects a tape during the DCB Tape Validation Exit, DFSMShsm never considers that it owned the tape and therefore never invokes the ARCTVEXT exit to return the tape to a scratch status.

Defining the environment for the tape management system

Figure 73 on page 226 shows the SETSYS commands that define a typical tape management system environment. Specify these commands in the ARCCMD*xx* PARMLIB member.

```
/***********************************************************************/
/*  SAMPLE SETSYS COMMANDS THAT DEFINE AN ENVIRONMENT FOR A            */
/*  TYPICAL TAPE MANAGEMENT SYSTEM.                                    */
/***********************************************************************/
/*
SETSYS CDSVERSIONBACKUP -
       BACKUPDEVICECATEGORY(TAPE)
SETSYS SELECTVOLUME(SCRATCH)
SETSYS TAPEDELETION(SCRATCHTAPE)
SETSYS TAPESECURITY(RACF)
SETSYS EXITON(ARCTVEXT)            /*  NOT NEEDED IF YOU ARE USING RMM */
/*
```

Figure 73. SETSYS Commands that Initialize the Environment for a Tape Management System

SETSYS SELECTVOLUME(SCRATCH)
The SCRATCH option of the SETSYS SELECTVOLUME command defines how initial and subsequent dump tapes are selected and how subsequent backup and migration tapes are selected.

SETSYS TAPEDELETION(SCRATCHTAPE)
The SCRATCHTAPE option of the SETSYS TAPEDELETION command directs DFSMShsm to remove empty tapes from DFSMShsm control. DFSMShsm notifies RMM of the tape's new status through an internal interface. If you are using another tape management product, you must request that DFSMShsm use the ARCTVEXT exit by invoking the SETSYS EXITON(TV) command. For more information about the ARCTVEXT exit, refer to *z/OS DFSMS Installation Exits*.

SETSYS TAPESECURITY(RACF)
The RACF option of the SETSYS TAPESECURITY command directs DFSMShsm to automatically add RACF protection to scratch tapes.

SETSYS EXITON(TV)

Programming Interface Information

The ARCTVEXT option of the SETSYS EXITON command directs DFSMShsm to invoke the tape volume exit, ARCTVEXT, when a DFSMShsm tape has been emptied. DFSMShsm releases control of a tape by performing a DELVOL function on the tape (also known as SETSYS TAPEDELETION processing).

When the ARCTVEXT exit is invoked, the code in the exit communicates to the tape management system whether the tape is no longer required by DFSMShsm and is eligible to be returned to the global scratch pool or is being returned to the specific scratch pool. The code for the exit is available from the vendors of the various tape management systems and, once the code has been installed into a LINKLIST library, the exit is invoked if you specify SETSYS EXITON(ARCTVEXT).

Note: You should only use the ARCTVEXT exit with non-IBM tape management systems, because DFSMShsm directly uses the DFSMSrmm interface (EDGTVEXT).

For more information about DFSMShsm tape management exits, see "Managing tapes with DFSMShsm installation exits" on page 227.

End Programming Interface Information

Data set naming conventions

Because some tape management systems recognize and associate only one high-level qualifier with DFSMShsm, ensure that you specify the same prefix for CDS-version backup, backup, and migration data sets. You can ensure that prefixes are the same when you specify the same name for the BACKUPPREFIX, MIGRATEPREFIX, and CDSVERSIONBACKUP parameters of the SETSYS command. If you do not specify the same prefix for CDS-version backup, backup, and migration data sets, only the prefix that matches the high-level qualifier defined to the tape management system is associated with DFSMShsm and excluded from management by the tape management system.

Attention: Changing BACKUPPREFIX or MIGRATIONPREFIX after the DFSMShsm environment is originally set, or having different BACKUPPREFIXs or MIGRATIONPREFIXs between hosts or sites could result in failures during recall, recover, and recycle processing.

Managing tapes with DFSMShsm installation exits

DFSMShsm provides the following exit points for tape management:

- ARCEDEXT (ABARS expiration date) exit is used with any tape management system that requires unique expiration dates.
- ARCTDEXT (tape data set) exit is used only when DFSMShsm co-manages tapes with a tape management system.
- ARCTVEXT (tape volume) exit lets any tape management system (other than DFSMSrmm) know that DFSMShsm is releasing ownership of a DFSMShsm tape; ARCTVEXT is only called when SETSYS EXITON(TV) has been specified.

Defining the device management policy for your site

DFSMShsm uses tape devices that read and write on reels of tape (3420, 3422, and 3430) or read and write on magnetic tape cartridges (3480, 3480X, 3490 and 3590-1). Your site's device management policy is determined by the choices you make for the following tasks:

- "Tape device selection"
- "Tape device conversion" on page 232
- "Specifying whether to suspend system activity for device allocations" on page 233
- "Specifying how long to allow for tape mounts" on page 235
- "Specifying the length of time before DFSMShsm takes action" on page 235

Tape device selection

If you have only one kind of tape device at your site, no consideration of tape device selection exists. For any tape-input or tape-output operation, only one kind of tape device can be selected.

However, if you have more than one kind of tape device at your site, a choice must be made as to which device is selected for a given function. You can direct MVS to select a device you specify, or you can allow MVS to choose the device for you. You can restrict MVS allocation to selecting the specific kind of tape device you want for a given function in either an SMS-managed tape library or in a nonlibrary environment. Restricting device selection can improve tape processing when high-performance devices are selected and can ensure that new devices are selected when they are introduced into your tape environment. For more information about restricting device selection, see "Restricting tape device selection."

Nonlibrary tape device selection

When DFSMShsm requests that a tape device be allocated and mounted, MVS allocates a tape device according to two aspects of the request:

- Whether the request is for a specific device type
- Whether the request is for a specific tape or for a scratch tape

MVS first honors the device type that DFSMShsm selects. If DFSMShsm specifies a generic unit name for a cartridge-type device, MVS allocates a cartridge-type device, but selects a cartridge-type device with or without a cartridge loader according to its own criteria. (If all the devices have cartridge loaders, the MVS criteria are unimportant.) If DFSMShsm specifies an esoteric unit name that is associated with devices with the cartridge loader, MVS allocates a device from that esoteric group.

If the DFSMShsm tape-allocation request is for a cartridge-type device, MVS selects a device by considering whether the request is for a specific tape or for a scratch tape. If the request is for a specific tape, MVS tries to allocate a device that *does not* have a cartridge loader. If the request is for a scratch tape, MVS tries to allocate a device that *does* have a cartridge loader. If the preferred kind of device is not available, MVS selects any available cartridge-type device.

Library tape device selection

Tape devices that are associated with SMS-managed tape libraries are selected based upon the technologies they support. You can restrict selection of these devices by specifying the data-class attributes that are associated with a device. For more information about data-class attributes, see "Define a data class" on page 193.

Note: When selecting empty tapes for output in an SMS-managed tape library, HSM selects the tape device, then locates a suitable tape for output. Because of this, environments that have multiple tape libraries assigned to the same storage group should use a global scratch queue instead of an HSM specific scratch pool. Using an HSM-specific tape pool in this environment will cause a scratch mount when only one library has empty tapes available for use and HSM allocates device in a different library. HSM does not pass volser or library name to the ACS routines so you cannot use ACS routines to prevent the problem.

Restricting tape device selection

When processing tapes, you can restrict device selection to certain devices as discussed in Table 28.

Table 28. Restricting Device Selection

Tape Environment	How to Restrict Device Selection
SMS-managed Tape Library	Restrict device allocation to a particular group of devices by associating them with a data class. For an example of a data class for an SMS-managed tape library, see "Define a data class" on page 193, and also refer to *z/OS DFSMS Implementing System-Managed Storage*. For general information about defining storage groups, refer to *z/OS DFSMSdfp Storage Administration*.

Table 28. Restricting Device Selection (continued)

Tape Environment	How to Restrict Device Selection

Nonlibrary	Restrict allocation to a particular group of tape devices by defining an esoteric unit name for them. An esoteric unit name, specified in the SETSYS USERUNITTABLE command, associates a group of similar devices with a name you define to MVS.
	• An esoteric group for 3590-1 tape devices must contain all 3590-1 tape devices of the same recording technology; for example, 128-track, 256-track, or 384-track recording capability.
	• An esoteric group for 3490 tape devices must contain all 3490 tape devices of the same recording technology. Emulated tape devices on different hardware must belong to the same category: 3590 Model B, 3590 Model E, 3590 Model H, VTS, or other.
	• An esoteric group for 3480 and 3480X devices can mix devices, but the esoteric group will not use improved data recording capability (IDRC). All devices associated with the esoteric unit name are required to have the 3480X feature if IDRC support is desired.
	
	A cartridge mounted at end-of-volume is assigned the esoteric unit name and compaction characteristics of the previous tape.
	
	Restrict allocation to a particular group of tape devices by specifying the *unittype* parameter of the following DFSMShsm tape output commands.
	ABACKUP *agname* UNIT(*unittype*)
	ARECOVER *agname* TARGETUNIT(*unittype*)
	DEFINE DUMPCLASS(*name*...UNIT(*unittype*))
	SETSYS
	  ARECOVERML2UNIT(*unittype*)
	  ABARSUNIT(*unittype*)
	  ARECOVERUNITNAME(*unittype*)
	  BACKUP(TAPE(*unittype*))
	  CDSVERSIONBACKUP
	    (BACKUPDEVICECATEGORY(TAPE
	    (UNITNAME(*unittype*))))
	  RECYCLEOUTPUT(BACKUP(*unittype*))
	  RECYCLEOUTPUT(MIGRATION(*unittype*))
	  SPILL(TAPE(*unittype*))
	  TAPEMIGRATION
	    (DIRECT(TAPE(*unittype*)))
	    (ML2TAPE(TAPE(*unittype*)))
	    (NONE(ROUTETOTAPE(*unittype*)))
	TAPECOPY ALTERNATEUNITNAME(*unittype1, unittype2*)
	TAPECOPY ALTERNATE3590UNITNAME *(unittype1, unittype2)*
	
	You can optionally substitute an esoteric unit name for the *unittype* parameter in the preceding list to restrict certain tape devices to DFSMShsm's use.

Note: The distinction in Table 28 on page 228 between SMS-managed and non-SMS-managed becomes apparent when your storage class ACS routine (based on inputs of data set name, unit type, and jobname) provides DFSMShsm with either a storage class name or null. When a storage class is provided (the request is for SMS-managed storage), the storage group ACS routine determines a storage group, and the data class ACS routine restricts selection within that storage group.

When the storage class ACS routine indicates "no storage class," the request is treated like the "Nonlibrary" case in Table 28 on page 228.

Optimizing cartridge loaders by restricting output to devices with cartridge loaders

DFSMShsm always requests a mount for a specific tape for input processing, so cartridge loaders are of little value for input. For mounting a single tape on a 3480 device type from which to recall or recover a data set, it may be more economical to allocate a device that *does not* have a cartridge loader.

To ensure that non-cartridge-loader devices are used for input, you can direct DFSMShsm to use esoteric unit names in a special way that directs a cartridge to be allocated on a different set of devices for input than was used for output. This activity is called *esoteric translation* and it occurs if:

- You have specified an esoteric unit name that is associated with a group of output devices.
- The output devices associated with the esoteric unit name are online when the SETSYS USERUNITTABLE command is specified. Esoteric translation is not dynamic and DFSMShsm knows only the devices that are online at the moment the SETSYS USERUNITTABLE command is issued.
- All devices associated with the esoteric unit name have cartridge loaders. For the exception, see "Removing ACL as a condition for D/T3480 esoteric unit name translation" on page 360.

Only if the preceding conditions have been met can esoteric translation occur. Esoteric translation occurs in either of two ways:

- If you specify the translation name, DFSMShsm retains the new unit type for input allocations of the tape.
- If you do not specify the translation name, DFSMShsm retains the generic unit type of the output esoteric for output allocations of the tape.

If you specify . . .	Then . . .
SETSYS USERUNITTABLE (ACLOUT:ACLIN)	And all of the preceding conditions have been satisfied, a tape is allocated on a different set of devices for input than was used for output.

Summary of esoteric translation results for various tape devices

Table 29 describes the headings and Table 30 on page 231 describes the esoteric translation results if a tape that is already known to DFSMShsm is selected for output. Ensure that the conditions described in "Optimizing cartridge loaders by restricting output to devices with cartridge loaders" have been satisfied.

Table 29. Legend for Esoteric Unit Name and Resulting Translation Occurring during Volume Selection

Heading	Description
ADDVOL DEVICE TYPE	Device name assigned to the tape with the ADDVOL command (generic or esoteric) prior to being used for output. The device name can be generic (for example, 3490, 3480X, 3480 or 3590-1) or the device name can be esoteric (a user-defined name).
HWC	SETSYS TAPEHARDWARECOMPACT setting

Table 29. Legend for Esoteric Unit Name and Resulting Translation Occurring during Volume Selection (continued)

Heading	Description
RESTRICTED UNIT	Output device restriction
ACL	That all devices in a given esoteric have the ACL feature
SETSYS USERUNITTABLE	SETSYS USERUNITTABLE setting
TRANSLATION RESULT	Device name that is associated with the tape after output. This device name is used for subsequent allocations of the tape for input. **ES3480 or ESI3480** Esoteric unit names that are associated with 3480 devices **ES3480X or ESI3480X** Esoteric unit names that are associated with 3480X devices **ES3490 or ESI3490** Esoteric unit names that are associated with 3490 devices **ES3590 or ESI3590** Esoteric unit names that are associated with 3590 devices **NA** Not applicable

Table 30. Specifying Esoteric Unit Name and Resulting Translation Occurring during Volume Selection

ADDVOL DEVICE TYPE	HWC Y	HWC N	RESTRICTED UNIT	ACL Y	ACL N	SETSYS USERUNITTABLE SETTING	TRANSLATION RESULT
3480	X		3480X	X		Not set	3480X
3480	X		3480X		X	Not Set	3480X
3480	X		ES3480X	X		(ES3480X) or (ES3480X:3480X)	3480X
						(ES3480X:ESI3480X)	ESI3480X
3480	NA	NA	3480	NA	NA	Not Set	3480
3480X	X		ES3480X		X	(ES3480X) or (ES3480X:3480X) or (ES3480X:ESI3480X)	ES3480X
ES3480	X		ES3480X	X		(ES3480X) or (ES3480X:3480X)	3480X
						(ES3480X:ESI3480X)	ESI3480X
3480		X	3480X	NA	NA	Not Set	3480
ES3480X		X	NA		X	(ES3480X) or (ES3480X:3480X) or (ES3480X:ESI3480X)	ES3480X (no change)
3480 3480X ES3480 ES3480X	NA	NA	3490	NA	NA	Not Set	3490
3480 3480X ES3480 ES3480X	NA	NA	ES3490	NA	NA	(ES3490) or (ES3490:3490)	3490
						(ES3490:ESI3490)	ESI3490
3490	NA	NA	ES3490	NA	NA	(ES3490) or (ES3490:3490) or (ES3490:ESI3490)	3490
							ESI3490
ES3490	NA	NA	NA	NA	NA	(ES3490) or (ES3490:3490)	3490
						(ES3490:ESI3490)	ESI3490

Table 30. Specifying Esoteric Unit Name and Resulting Translation Occurring during Volume Selection (continued)

ADDVOL DEVICE TYPE	HWC		RESTRICTED UNIT	ACL		SETSYS USERUNITTABLE SETTING	TRANSLATION RESULT
	Y	N		Y	N		
3590-1	NA	NA	ES3590	NA	NA	(ES3590) or (ES3590:3590-1) or (ES3590:ESI3590)	3590-1
							ESI3590
ES3590	NA	NA	NA	NA	NA	(ES3590) or (ES3590:3590-1)	3590-1
						(ES3590:ESI3590)	ESI3590

Tape device conversion

Converting to new device types often requires moving data from existing cartridges to new cartridges. You can direct data to new devices by specifying output unit type restrictions for RECYCLE. You can limit the input to RECYCLE by specifying a volser range. Table 31 describes how to convert your existing device types to new device types.

Table 31. How to Convert to Different Device Types

Tape Environment	How to Convert to New Device Types
SMS-Managed Tape Library	DFSMShsm selects new devices in a tape library when you update the data class and ACS routines to select the devices you want.
Nonlibrary	DFSMShsm selects new devices in a nonlibrary environment when you update the *unittype* variable of the commands shown in Table 28 on page 228.

Reading existing data on new device types after a device conversion in a nonlibrary environment

The following discussion describes what DFSMShsm does to enable existing data to be read on new device types after a device conversion outside of an SMS-managed tape library.

Converting from 3480 to 3490 devices in a nonlibrary environment: If the system support for 3490 devices is installed, DFSMShsm converts the following generic unit names to the special esoteric names provided by the system. The esoteric unit name can be for a device other than the 3480, 3480X, or 3490.

- 3480 (used for output) is changed to SYS3480R for input drive selection. SYS3480R is a special esoteric name that is associated with all 3480, 3480X, and 3490 devices. Any device in this esoteric is capable of reading a cartridge written by a 3480 device.
- 3480X (used for output) is changed to SYS348XR for input drive selection. SYS348XR is a special esoteric name that is associated with all 3480X and 3490 devices. Any device in this esoteric is capable of reading a cartridge written by a 3480X device.

In a JES3 system, you need to define the SYS3480R and SYS348XR esoterics during JES3 initialization. SMS allocation performs this conversion in a library environment. A 3490 device can read data written on 3480, 3480X, or 3490 devices. A 3480X device can read data written on 3480 or 3480X devices.

Converting from 3420 to 3480 devices in a nonlibrary environment: If you reuse an esoteric unit name for incompatible devices (for example, the data was written when the esoteric unit name was for a group of 3420 device types, but the esoteric unit name is now used for a group of 3480 devices), DFSMShsm detects the incompatibility of the tape and the device type and replaces the reused esoteric unit name with the generic name of the old device.

Specifying whether to suspend system activity for device allocations

Sometimes DFSMShsm requests a device allocation and that request cannot be met because all tape devices are temporarily unavailable. When no tape devices are immediately available, dynamic allocation can wait for a tape device to become available before it returns to DFSMShsm or it can retry the request at a later time. If dynamic allocation waits for a device, all DFSMShsm processing soon stops until a device is allocated. If dynamic allocation does not wait and DFSMShsm retries the request later, other DFSMShsm functions can continue processing.

The WAIT and NOWAIT options of the tape device-allocation parameters (INPUTTAPEALLOCATION, OUTPUTTAPEALLOCATION, and RECYCLETAPEALLOCATION) control whether all DFSMShsm processing is suspended for a device allocation or not. The options you should use depend on:

- Whether you are managing tapes in a JES2 or a JES3 environment
- The number of tape devices that are available for DFSMShsm processing
- How much help you provide in making tape devices available

  You can make tape devices available by restricting the use of tape devices. For more information about restricting output to devices with similar characteristics, see "Restricting tape device selection" on page 228.

Note: The WAIT|NOWAIT option does not apply to the aggregate-backup-and-recovery support (ABARS) secondary address space. For information about specifying a wait option for ABARS, see "Enabling ABARS ABACKUP and ARECOVER to wait for a tape unit allocation" on page 339.

Specifying the WAIT option

When DFSMShsm requests a tape allocation, an exclusive enqueue is obtained on the task input/output table resource (SYSZTIOT). This exclusive enqueue stays in effect until the tape device is allocated or the request fails. Therefore, if you specify the WAIT option, all DFSMShsm functions are suspended until a device is allocated to the task that holds the exclusive enqueue.

The WAIT option is recommended only for:

- Environments that use JES3 to manage tapes.

  Because of main device scheduling, you should specify INPUTTAPEALLOCATION(WAIT) so that the DFSMShsm allocation request can be put into the queue for device availability. Input tape requests that occur during the day need to be honored, because they will usually be requests to recall or recover data sets.

  You may want to use OUTPUTTAPEALLOCATION(WAIT) if you are doing interval migration with some data sets migrating directly to tape in a JES3 environment. Interval migration runs on an hourly schedule throughout the day, and you want DFSMShsm to be able to obtain a tape device through the main device scheduler. For more information about interval migration, see *z/OS DFSMShsm Storage Administration*.
- Environments that use esoteric unit names to restrict the use of some tape devices to DFSMShsm processing. You can use the WAIT option because a tape device should usually be available.

If you specify . . .	Then . . .
SETSYS INPUTTAPEALLOCATION (WAIT) SETSYS OUTPUTTAPEALLOCATION (WAIT) SETSYS RECYCLETAPEALLOCATION (WAIT)	The dynamic-allocation function of MVS waits for a tape device allocation before returning control to DFSMShsm.

Note: Ensure that there are enough tape devices available for the number of tasks currently running. What can be perceived as a DFSMShsm-not-running condition can occur because of the exclusive enqueue that MVS puts on SYSZTIOT while awaiting completion of tape device allocation.

Specifying the NOWAIT option

When you specify the NOWAIT option, dynamic allocation does not wait for a tape device to become available. Dynamic allocation returns control to DFSMShsm immediately, either with a tape device allocated or with a failure indication if no tape device is immediately available. Therefore, other DFSMShsm data-movement functions can continue to process data while the task requiring tape is waiting to reissue the allocation request. The NOWAIT option is recommended for:

- JES2 environments.

  Because a JES2 system does not preschedule devices, DFSMShsm has an opportunity in such a system to obtain the needed tape devices when it requests them.
- Environments that hold some tape devices offline so the operator can vary the devices online when DFSMShsm issues the ARC0381A message.

If you specify . . .	Then . . .
SETSYS INPUTTAPEALLOCATION (NOWAIT) SETSYS OUTPUTTAPEALLOCATION (NOWAIT) SETSYS RECYCLETAPEALLOCATION (NOWAIT)	The dynamic-allocation function of MVS returns control to DFSMShsm and does not suspend system activity while a device is being allocated.

If the allocation request fails because no devices are available, DFSMShsm repeats the request six more times. If a tape device cannot be allocated after seven tries, DFSMShsm issues a message asking the operator either to cancel the request or to repeat the set of requests.

If a tape device can become available within a reasonable time, the operator should reply with RETRY. DFSMShsm then repeats the request seven more times. If a tape device is still not allocated, DFSMShsm again asks the operator whether it should cancel the request or repeat the request. A reply of CANCEL fails the DFSMShsm task immediately.

At some sites, some device addresses are defined to the system only as spare addresses and not as actual devices. When DFSMShsm issues a tape allocation request and all tape devices are in use, the allocation request is not failed with a reason of no units available. Instead, the allocation message (IEF238D) is issued, and the only tape device addresses listed are the spare addresses. If the device addresses are only spares, the operator should reply CANCEL to this message. If this is the first time the operator has replied CANCEL to message IEF238D, DFSMShsm issues an ARC0381A message to verify the request that it discontinue trying to find a target tape device.

Specifying how long to allow for tape mounts

The MOUNTWAITTIME option determines the maximum time that DFSMShsm waits for a tape mount before it issues a message to the operator. The MOUNTWAITTIME parameter is assigned a numeric value that specifies a reasonable amount of time to mount the tape. There are three mount processes: mounts by humans, robotic mounts, and virtual mounts. There is a wide range of the amounts of time for each mount process, and the MOUNTWAITTIME option is applied independently of which mount process is occurring.

Specifying the tape mount parameters

When DFSMShsm dynamically allocates a tape device, dynamic allocation does not issue a mount message. Later, OPEN processing issues the mount message (IEC501A) during a shared enqueue on the task input/output table (SYSZTIOT) resource. The shared enqueue is preferred over the exclusive enqueue because multiple functions can process while awaiting a tape mount.

Specifying the length of time before DFSMShsm takes action: To guard against an unrecoverable error, such as a lost tape, an unloadable tape, an overloaded virtual tape subsystem, or a distracted operator, DFSMShsm allows you to specify a maximum time to wait for a tape mount.

If you specify . . .	Then . . .
SETSYS MOUNTWAITTIME(15)	DFSMShsm sets a timer each time a mount request is issued.

The value set into the timer is specified with the SETSYS MOUNTWAITTIME command. If the timer expires before the tape is mounted or if the system timer is inoperative, DFSMShsm sends the ARC0310A operator message asking if the tape can be mounted. If the operator replies Y, DFSMShsm sets a second timer of the same length (unless the system timer is inoperative). If the second timer expires before the tape is mounted, DFSMShsm marks the tape as unavailable and selects another output tape. If the operator replies N:

- The DFSMShsm backup, migration, and recycle functions mark the requested tape as unavailable and select another output tape.
- The DFSMShsm dump function removes the tape from the selection list and fails the volume-dump operation.
- The DFSMShsm recovery, restore, and recall functions fail.

For *input* tapes, DFSMShsm sets the timer only for the first tape requested for any particular recall, recover, or recycle. If the data set being recalled, recovered, or recycled spans more than one tape, the mount requests for all tapes after the first tape are not protected by the MOUNTWAITTIME timer. Thus, if succeeding tapes after the first are not available, the task can be neither continued nor canceled until a correctly labeled tape is mounted.

The MOUNTWAITTIME timer protects all requests for *output* tapes.

The MOUNTWAITTIME is not used for aggregate processing since its waiting does not greatly affect the DFSMShsm primary address space processing.

Implementing the performance management policies for your site

Your site's performance management policy is determined by the choices you make for the following tasks:

- "Reducing tape mounts with tape mount management"
- "Doubling storage capacity with extended high performance cartridge tape" on page 237
- "Defining the environment for enhanced capacity and extended high performance cartridge system tape" on page 237
- "Specifying how much of a tape DFSMShsm uses" on page 240
- "Implementing partially unattended operation with cartridge loaders in a nonlibrary environment" on page 242
- "Improving device performance with hardware compaction algorithms" on page 244
- "Creating concurrent tapes for on-site and offsite storage" on page 245
- "Considerations for duplicating backup tapes" on page 247

Reducing tape mounts with tape mount management

Significant performance improvements to output tape processing can be achieved by optimizing tape usage with the *tape mount management* methodology. Tape mount management (TMM) provides tape performance enhancements that:

- Reduce tape mounts

  TMM methodology redirects new site-identified data sets to a specific storage group with the expectation that they will be migrated to tape when the amount of data warrants a tape mount.
- Reduce tape inventory and maximize media utilization

  TMM candidates can be written by DFSMShsm to tape in single file, compacted form with a single tape mount. DFSMShsm attempts to fill the entire tape before selecting another tape.
- Improve turnaround time for batch jobs

  Batch jobs using data sets that are written to the system-managed DASD buffer do not wait for tape mounts and can perform I/O at DASD or cache speeds.

Refer to *z/OS DFSMS Implementing System-Managed Storage*, for more information about tape mount management.

Doubling storage capacity with enhanced capacity cartridge system tape

Enhanced capacity cartridge system tape is a special 3490 tape cartridge that can be used only on 36-track 3490 tape devices. You can write twice as much data on this tape as you can on a standard cartridge system tape.

The differences between enhanced capacity cartridge system tapes and standard cartridge system tapes do not present a concern unless you are copying the contents of one tape to another in a tape environment that mixes media (enhanced capacity cartridge system tapes and standard cartridge system tapes).

During TAPECOPY processing, if standard 3490 tapes are mounted when enhanced capacity tapes are needed, or vice versa, processing failures occur. You can avoid these failures by directing DFSMShsm to issue messages to the operator indicating whether standard tapes or enhanced capacity tapes should be mounted during TAPECOPY processing.

Doubling storage capacity with extended high performance cartridge tape

Extended high performance cartridge tape is a special 3590 tape cartridge that can be used only on 3590-1 tape devices. You can write twice as much data on these tapes as you can on standard 3590-1 high performance cartridge tapes.

The differences between extended high performance cartridge tapes and standard cartridge tapes do not present a concern unless you are copying the contents of one tape to another in a tape environment that mixes media (extended high performance cartridge tapes and standard cartridge tapes).

During TAPECOPY processing, if standard 3590-1 tapes are mounted when extended high performance tapes are needed, or vice versa, processing failures occur. You can avoid these failures by directing DFSMShsm to issue messages to the operator indicating whether standard tapes or enhanced capacity tapes should be mounted during TAPECOPY processing.

Defining the environment for enhanced capacity and extended high performance cartridge system tape

The SETSYS TAPEOUTPUTPROMPT command determines whether messages are sent to the operator indicating the kind of tape to mount for TAPECOPY processing. You need issue this command only in a non-SMS-managed tape environment. If you are processing tapes in an SMS-managed tape library, the robot and MVS communicate directly and no operator intervention is required.

If you specify . . .	Then . . .
SETSYS TAPEOUTPUTPROMPT (TAPECOPY) or SETSYS TAPEOUTPUTPROMPT (TAPECOPY(YES)),	DFSMShsm sends prompting messages to the operator indicating which kind of 3490 tape should be mounted.
SETSYS TAPEOUTPUTPROMP (TAPECOPY(NO))	DFSMShsm reverts to not sending prompting messages.

You can specify the TAPECOPY ALTERNATEUNITNAME command to specify two esoteric unit names; the first name is used if a standard 3490 tape is required for TAPECOPY processing, and the second name is used if an enhanced capacity tape is required. Then you can preload the two different cartridge types into the devices associated with the two different esoteric unit names.

If you specify . . .	Then . . .
TAPECOPY ALL ALTERNATEUNITNAME (TAPESHOR,TAPELONG)	DFSMShsm copies 3490 backup and migration data sets that reside on standard cartridge system tapes to devices in the esoteric named TAPESHOR. DFSMShsm also copies 3480 backup and migration data sets that reside on enhanced capacity cartridge system tape to devices in the esoteric named TAPELONG. DFSMShsm uses the first esoteric unit name that you specify in the ALTERNATEUNITNAME parameter for standard tape cartridges and the second esoteric unit name for extended capacity tape cartridges.

You can specify the TAPECOPY ALTERNATE3590UNITNAME command to specify two esoteric unit names; the first name is used if a standard 3590 tape is required for TAPECOPY processing. The second name is used if a high performance cartridge tape is required. Then you can preload the two different cartridge types into the devices that are associated with the two different esoteric unit names.

If you specify . . .	Then . . .
TAPECOPY ALL ALTERNATE3590UNITNAME (TAPE5S,TAPE5L)	DFSMShsm copies 3590 backup and migration data sets that reside on standard high performance cartridge tapes to devices in the esoteric named TAPE5S. DFSMShsm also copies backup and migration data sets that reside on extended high performance cartridge tape to devices in the esoteric named TAPE5L.

For more information about defining esoteric unit names, see "Restricting tape device selection" on page 228. For more information about the TAPEOUTPUTPROMPT parameter of the SETSYS command and the ALTERNATEUNITNAME parameter of the TAPECOPY command, see *z/OS DFSMShsm Storage Administration*.

Utilizing the capacity of IBM tape drives that emulate IBM 3490 tape drives

The IBM capacity utilization and performance enhancement (CUPE) allows DFSMShsm to more fully utilize the high-capacity tapes on IBM tape drives that emulate IBM 3490 tape drives.

You can select to more fully utilize tapes on CAPACITYMODE switchable IBM 3590 tape drives that emulate IBM 3490 tape drives, or to stop filling tapes at a point that is compatible with non-CAPACITYMODE switchable IBM 3590 tape drives. The IBM 3592 Model J is also CAPACITYMODE switchable when it is emulating a 3490 tape drive. Unless there is some known compatibility issue, these drives should be run with CAPACITYMODE(EXTENDED).

^ You can use the 3592 Model E05 and newer tape drives only in 3590 emulation mode; never 3490. The 3592 Model J1A can operate in 3490 emulation mode only when using MEDIA5 for output.

Defining the environment for utilizing the capacity of IBM tape drives that emulate IBM 3490 tape drives

To define the environment to use the capacity of IBM tape drives that emulate IBM 3490 tape drives, you must perform the following steps:

1. Define an esoteric to MVS that only contains IBM tape drives emulating IBM 3490 tape drives with the extended capacity support (CAPACITYMODE switchable). All of the drives within an esoteric must use the same recording technology.
2. Vary online the drives that are CAPACITYMODE switchable after any IPL.
3. Use the SETSYS USERUNITTABLE command to define the new esoteric to DFSMShsm. Be sure to provide an input translation unit so a non-CAPACITYMODE switchable unit is not selected by default.

   Example: When you specify the SETSYS USERUNITTABLE(*es1*, *es2out:es2in*,***es3cupe:es3cupe***) command, the esoteric unit name ***es3cupe*** is identified to DFSMShsm as valid. Since the same esoteric unit name is specified for both input and output, the same units are candidates for input and output allocations. Substitute your esoteric name for ***es3cupe***.

4. Use the SETSYS TAPEUTILIZATION(CAPACITYMODE(EXTENDED | COMPATIBILITY) UNITTYPE(*new esoteric*) PERCENTFULL(*nn*)) command to tell DFSMShsm how to use the drives in that esoteric.
5. Use the appropriate DFSMShsm commands to define for output, the new esoteric to DFSMShsm functions, like BACKUP or MIGRATION.

   Example: You can use the SETSYS backup command for backup functions or the SETSYS TAPEMIGRATION command for migration.

^ **Note:** You can only use the 3592 Model E05 and newer tape drives only in 3590 emulation mode; never 3490. The 3592 Model J1A can operate in 3490 emulation mode only when using MEDIA5 for output.

Tape selection: When a DFSMShsm function uses a tape drive to write output, DFSMShsm selects either an empty tape or a partially filled tape cartridge. For IBM 3590 or 3592 tape drives that emulate IBM 3490 tape drives to write data to partial tapes, the CAPACITYMODE of the tape must match the CAPACITYMODE of the tape drive. If your installation has set the output esoteric to use CAPACITYMODE(EXTENDED) and only CAPACITYMODE(COMPATIBILITY) partial tapes are available, an empty tape is mounted.

Drive selection: For IBM 3590 tape drives that emulate IBM 3490 tape drives to read data, data written in CAPACITYMODE(COMPATIBILITY) can be read by any device capable of mounting the tape. Data written in CAPACITYMODE(EXTENDED) requires a CAPACITYMODE switchable esoteric device. Thus, there may be idle emulated IBM 3590 tape drives while an DFSMShsm function is delayed for a lack of a drive in a CAPACITYMODE switchable esoteric unit.

Remote site compatibility: When you are using tapes at a remote site, keep in mind that the remote site must utilize all data that is written by the home site. Data that is written to tapes with CAPACITYMODE(EXTENDED) specified, requires that there be an esoteric unit defined at the remote site with only CAPACITYMODE switchable drives.

Guideline: Until this environment is established at both locations, use drives that are capable of operating in a CAPACITYMODE(COMPATIBILITY) environment.

The SELECT (CAPACITYMODE(EXTENDED| COMPATIBILITY) option of the LIST TTOC command lists the CAPACITYMODE characteristics of tapes that are written in an HSMplex.

Data created at the home site on esoteric units with CAPACITYMODE(COMPATIBILITY) specified can be read on esoteric units at the remote site that have CAPACITYMODE(EXTENDED) specified. The tapes that contain this data are not selected for output by esoteric units that have CAPACITYMODE(EXTENDED) specified while they still contain CAPACITYMODE(COMPATIBILITY) data.

The ABACKUP function does not write data in CAPACITYMODE (EXTENDED). If user tapes that were created with CAPACITYMODE(EXTENDED) specified accompany the ABACKUP data, tape drives that are CAPACITYMODE switchable will be needed at the ARECOVER site to read them.

IBM 3590 capacity utilization considerations

Allowing IBM 3590 tape drives that emulate IBM 3490 tape drives to operate in COMPATIBILITY mode makes coexistence possible in an HSMplex until you have updated all of your systems. The use of the SETSYS TAPEUTILIZATION(CAPACITYMODE) parameter makes this possible.

If you install this IBM 3590 CUPE support on one DFSMShsm at a time, run the supported hosts with CAPACITYMODE(COMPATIBILITY) specified until you install the IBM 3590 CUPE support across the entire HSMplex. Then all or any combination of hosts in the HSMplex can run with CAPACITYMODE(EXTENDED) specified.

Note: Recall and recover fail with an error message when any DFSMShsm task in a system that does not have access to an esoteric with CAPACITYMODE switchable drives and the data set is on a CAPACITYMODE(EXTENDED) tape. Any system running with CUPE coexistence support fails the recall or recover function when a needed data set is on a CAPACITYMODE(EXTENDED) tape.

In order for DFSMShsm to recognize that a drive has the CAPACITYMODE(EXTENDED) function, that drive must have been online at least once since IPL. This is a new requirement. The system programmer might chose to cause all drives to be online at IPL or to issue VARY ONLINE commands automatically.

High capacity tapes are more fully utilized when you specify the CAPACITYMODE(EXTENDED) parameter for that defined esoteric. If a DFSMShsm function needs to mount a tape that was created with CAPACITYMODE(EXTENDED) specified but no CAPACITYMODE switchable drives are available, the DFSMShsm function fails with an error message. MEDIA3 and MEDIA4 tapes that are filled when the CAPACITYMODE(COMPATIBILITY) parameter is specified, can be read by all IBM 3590 drives emulating IBM 3490 drives regardless of the CAPACITYMODE switchable capability as long as the recording technologies are read compatible.

Rule: To return to a CAPACITYMODE(COMPATIBILITY) environment, you must reissue the SETSYS TAPEUTILIZATION(CAPACITYMODE(COMPATIBILITY)) command so that no new data is written in CAPACITYMODE(EXTENDED) format.

Specifying how much of a tape DFSMShsm uses

For TAPECOPY command or DUPLEX option: If you are copying the contents of one tape to another with the TAPECOPY command or are using the concurrent creation option DUPLEX, you need to be aware of minor inconsistencies that can exist in the length of cartridge-type tapes. Because the TAPECOPY command copies the entire contents of one tape to another, it is important that enough media is available to copy the entire source tape to its target. Therefore, when you are copying tapes with the TAPECOPY command, use the default options (the equivalent of specifying the TAPEUTILIZATION command with the PERCENTFULL option of 97 percent). DFSMShsm marks the end of volume when tapes are 97 percent full. When you use the duplex option, it is recommended that you use the value 97% to ensure that you can write the same amount of data to both tapes. During duplexing, the NOLIMIT parameter of TAPEUTILIZATION will be converted to the default of 97%.

If you are not copying tapes with the TAPECOPY command and you are not creating two tapes using the DUPLEX option, you can specify the TAPEUTILIZATION command with a PERCENTFULL option of 100%.

Tape utilization is specified differently for library environments than for nonlibrary environments. Table 32 shows how tape utilization is specified for both library and nonlibrary environments.

Table 32. Specifying Tape Utilization. Tape utilization is specified differently for library and nonlibrary environments. In this example, DFSMShsm marks end of volume when the tape is 97% full.

Tape Environment	Specifying Tape Utilization
Nonlibrary	SETSYS TAPEUTILIZATION(UNITTYPE(3590-1) PERCENTFULL(97))
SMS-Managed Tape Library	SETSYS TAPEUTILIZATION(LIBRARYBACKUP PERCENTFULL(97)) or SETSYS TAPEUTILIZATION(LIBRARYMIGRATION - PERCENTFULL(97))

Tip: You can substitute an esoteric unit name for the generic device name shown in the preceding example.

Note: Virtual tape systems should generally use a PERCENTFULL value of 97% unless a bigger value is needed to account for virtual tapes larger than the nominal 400 MB standard capacity MEDIA1 or 800 MB enhanced capacity MEDIA2 tapes. In the case of the newer virtual tape systems (TS7700 Release 1.4 and above), where DFSMShsm derives media capacity by checking the mounted virtual tape, DFSMShsm allows a PERCENTFULL value up to 110%. Anything larger is reduced to 100%. For older virtual tape systems, where DFSMShsm cannot dynamically determine virtual tape capacity, PERCENTFULL values larger than 110% are honored. For a detailed description of the SETSYS TAPEUTILIZATION command, see *z/OS DFSMShsm Storage Administration*.

For 3590 Model E devices: If you are using IBM 3590-E*x* models in either native or 3490 emulation mode, DFSMShsm defaults to writing 97% of the true cartridge’s capacity and you probably do not need to specify TAPEUTILIZATION for those output devices. If you are using IBM 3590-B*x* drives (128-track recording) to emulate 3490 devices, you need to specify TAPEUTILIZATION PERCENTFULL of a much larger value than the default. The recommended value is 2200, which instructs DFSMShsm to use 2200% of the logical cartridge’s capacity as the capacity of tapes that are written to the supplied unit name. For a detailed discussion of the SETSYS TAPEUTILIZATION command, see *z/OS DFSMShsm Storage Administration*.

Note: For 3490 emulation by other hardware vendors, check with the vendors to determine which percentage should be specified.

For 3592 devices: Customers requiring very fast access to data on a MEDIA5 , MEDIA9, or MEDIA11 tape can exploit the 3592 performance scaling feature. Within DFSMShsm, performance scaling applies to single file tape data set names in both tape libraries and stand-alone environments.

Through the ISMF Data Class application, the performance scaling option uses 20% of the physical space on each tape and keeps all data sets closer together and closer to the initial load point. Performance scaling also allows the same amount of data to exist on a larger number of tapes, so more input tasks can run at the same time. This can increase the effective bandwidth during operations that read data from the tape. Alternatively, performance segmentation allows the use of most of the physical media while enhancing performance in the first and last portions of the tape.

As an alternative to using MEDIA5 , MEDIA9, or MEDIA11 tapes with performance scaling, consider using MEDIA7 or MEDIA13 tapes for high performance functions. Your MEDIA5 , MEDIA9, or MEDIA11 tapes could then be used to their full capacity.

Related reading

- For more information about the 3592 tape drives and the DFSMS software needed to use them, see *z/OS DFSMS Software Support for IBM System Storage TS1140, TS1130, and TS1120 Tape Drives (3592)*.

Implementing partially unattended operation with cartridge loaders in a nonlibrary environment

Maximum automation of your tape processing environment is possible when you process tapes in an SMS-managed tape library (see "SMS-managed tape libraries" on page 192); however, you can automate your tape processing in a non-SMS-managed tape environment by setting up cartridge loaders for unattended operation.

The cartridge loader, a hardware feature of 3480, 3480X, 3490, and 3590 devices, provides a non-SMS-managed tape environment with the ability to partially automate tape processing by allowing the operator to pre-mount multiple output tapes for migration, backup, or dump processing. By defining a global scratch pool, and by marking partially filled tapes full, you create an environment where migration or backup data sets or a dump copy are mounted and written to without operator intervention.

The allocation of cartridge-type devices is affected by:

- The cartridge loader switch settings that you select
- The scratch tape pool
- The way that DFSMShsm selects the first output tape (initial selection) and selects the tapes to continue with (subsequent selection)
- The way that MVS selects devices for allocation
- The way that you restrict device selection
- The way that you handle tapes that are partially full
- The way that DFSMShsm manages empty tapes
- The way that DFSMShsm manages the protection for empty tapes

Defining the environment for partially unattended operation

The following sequence of steps establishes the environment for implementing unattended operations with cartridge loaders:

1. Set the Mode Selection switch on the cartridge loader to system mode. System mode enables the cartridge loader to function differently for specific and nonspecific tape-mount requests. System mode also causes the cartridge loader to wait for scratch tape mounts before it inserts a cartridge into the drive.

2. The global scratch pool optimizes unattended operation. Ensure that you have specified the following commands in your ARCCMD*xx* PARMLIB member to fully utilize cartridge loaders:
   - SETSYS SELECTVOLUME (SCRATCH)
   - SETSYS PARTIALTAPE(MARKFULL)
   - SETSYS TAPEDELETION(SCRATCH)

   Table 33 describes the result of each of these SETSYS commands.

Table 33. Defining the Cartridge-Loader Environment

SETSYS Command	Result
SETSYS SELECTVOLUME (SCRATCH)	Creates a tape processing environment that enables most DFSMShsm output tapes to be scratch and hence enables unattended-output operation. By doing so, you direct DFSMShsm to issue nonspecific volume (PRIVAT) mount requests when it encounters an end-of-volume condition during tape output processing. The mount PRIVAT request causes the cartridge loader to mount the next cartridge automatically. For more information about defining scratch pools, see "Selecting a scratch pool environment" on page 211.
SETSYS PARTIALTAPE (MARKFULL)	Directs DFSMShsm to mark partially filled tapes full. Even though there may be additional room on the tape, by marking the tape full, DFSMShsm enables a scratch tape to be selected for the initial tape the next time the same function begins. The MARKFULL option enables the system to be unattended at the time the function begins; the REUSE option requires an operator to be present to mount the initial tape for each task of the function, therefore, enabling unattended operation only after the initial tape for each task has been mounted. For more information about marking tapes full, see "Selecting a scratch pool environment" on page 211.
SETSYS TAPEDELETION (SCRATCHTAPE)	Directs DFSMShsm to release recycled tapes to the global scratch pool and delete its records of them. For more information about returning empty tapes to the scratch pool, see "Selecting a scratch pool environment" on page 211.

3. Ensure that tape devices with cartridge loaders are allocated for DFSMShsm tape requests.

   You should define esoteric unit names to restrict certain tape devices to DFSMShsm's use, at least for the time that the DFSMShsm function runs. You could use different esoteric unit names for different DFSMShsm functions. If only some of your cartridge-type tape devices have cartridge loaders, define an esoteric unit name to MVS so that there is a device name that can restrict a DFSMShsm function to selecting only tape devices that have cartridge loaders.

   If you have defined an esoteric unit name for 3480 or 3490 devices that have cartridge loaders and if you specify the esoteric unit name in the *unittype* subparameter, DFSMShsm requests output scratch tape mounts only for the esoteric unit name, and MVS allocates only devices that have the cartridge loader installed. (For more information about esoteric unit names, see the USERUNITTABLE parameter of the SETSYS command in *z/OS DFSMShsm Storage Administration*.) For more information about defining esoteric unit names, see "Restricting tape device selection" on page 228.
4. Ensure that empty tapes can be returned to scratch status.

   If you are using a tape management system other than DFSMSrmm, ensure that you have specified SETSYS EXITON(TV) to invoke the tape volume exit so that tapes can be returned to the global scratch pool. For more information about coordinating DFSMShsm and a tape management system, see "Communicating with the tape management system" on page 225.

If you are not using a tape management system, ensure that you have specified SETSYS TAPESECURITY(RACF|RACFINCLUDE) to make tapes immediately usable when they are automatically returned to the scratch pool. For more information about removing protection from empty tapes, see "Removing the security on tapes returning to the scratch pool" on page 225.

5. Do not make any volumes known to DFSMShsm with the ADDVOL command. The operator then can load scratch tapes into all of the tape devices that DFSMShsm selects and can leave the tape devices unattended while DFSMShsm performs its output functions.

Improving device performance with hardware compaction algorithms

Data compaction improves tape processing performance because tapes can hold more data when the data is compacted, thus reducing the number of tape mounts when tape media is fully utilized. DFSMShsm provides software compaction, but that compaction depends on processor resources to compact data.

Data compaction can also be implemented at the device level, allowing the hardware, and not the main processor, to compact data. These kinds of compaction algorithms are available on all 3490 and 3590-1 devices and can be installed on 3480 devices.

Note: A 3480 device with improved data-recording capability (IDRC) is referred as a 3480X and the tape that is compacted with a 3480X device as a 3480X tape.

The cartridge-type device with the compaction algorithm function compacts data whenever data is moving to tape, whether it is from a level 0 volume, DASD migration volume, or another tape. When data is subsequently read from a cartridge-type device, data is decompacted before DFSMShsm receives the data.

Specifying compaction for non-SMS-managed tape library data

Table 34 shows the compaction status assigned to an empty tape when it is first selected for output.

Table 34. Tape Device Hardware-Compaction Capability

SETSYS Parameter	Output Device (3480)	Output Device (3480X)	Output Device (3490)	Output Device (3590-1)
TAPEHARDWARE COMPACT	No Compaction	Compaction	Compaction	Compaction
NOTAPEHARDWARE COMPACT	No Compaction	No Compaction	Compaction	Compaction

As you can see, the TAPEHARDWARECOMPACT|NOTAPEHARDWARECOMPACT parameters of the SETSYS command apply only to 3480X devices; 3480 devices cannot compact data and 3490 and 3590 devices always compact data when it is written by DFSMShsm unless overridden by a data class option.

For hardware compaction of data, it is recommended that you turn off DFSMShsm software compaction when the hardware compaction algorithm function is available and when only one step is involved in moving a data set from level 0 directly to tape.

If two steps are involved, such as migration to ML1 DASD followed by movement to tape, the decision is more involved. You need to determine the space-saving benefit on level 1 volumes and the time the data resides there, versus the processing-unit usage (CPU time) to do the compaction. You might consider specifying compaction but use the ARCMDEXT installation exit to *not* compact data sets that are either small or moved quickly on to migration level 2. Using compaction algorithms on data already compacted by DFSMShsm does not, in general, significantly help or hinder. For more information about using the ARCMDEXT exit, refer to *z/OS DFSMS Installation Exits*.

Specifying compaction for tape library data

If you are managing your tapes within an SMS-managed tape library, compaction of data is controlled by the data class associated with the data sets on the tape. If the assigned data class does not specify the COMPACTION attribute or no data class is assigned, the compaction is controlled with the DEVSUP*nn* PARMLIB member.

Because library data is SMS-managed and because SMS attributes are determined by the constructs that define SMS-managed data, compaction of data on library devices is determined by the data class assigned to the data set. For more information about data compaction in a library environment, refer to *z/OS DFSMS Implementing System-Managed Storage*.

Creating concurrent tapes for on-site and offsite storage

The DFSMShsm duplex tape function provides an alternative to TAPECOPY processing for backup and migration cartridge tapes. This option allows you to create two tapes concurrently: the original tape can be kept on-site while the alternate tape can be taken offsite or written to a remote tape library. The pair of tapes always maintain an “original versus alternate” distinction. The original cartridge-type tape volume will have one of the following data set names:

```
prefix.HMIGTAPE.DATASET
prefix.BACKTAPE.DATASET
```

while the alternate cartridge-type tape volume will have one of the following data set names:

```
prefix.COPY.HMIGTAPE.DATASET
prefix.COPY.BACKTAPE.DATASET
```

These data set name formats allow the new tapes to remain compatible with the current tapes created by the TAPECOPY command.

Within an **SMS environment**, ACS routines can direct the alternate tape to a different tape library, such as one at an offsite location. Within a **non-SMS environment**, the output restricter (for example, SETSYS UNIT(3590)) is used for both the original and the alternate. If allocation routing separation is needed, it must be done outside of DFSMShsm. Alternate tapes must keep the same tape geometry as the original tape (for example, both must be 3590 standard length tapes). For those customers who are drive-constrained, DFSMShsm maintains the existing TAPECOPY creation methods.

Duplex tape creation

To specify duplexing for backup and migration volumes, issue the SETSYS command with the keyword DUPLEX. BACKUP is an optional subparameter that specifies whether duplex alternates will be made for backup volumes. MIGRATION is another optional subparameter that specifies whether duplex alternates will be made for migration volumes. You can also specify that duplexing occurs for both backup and migration volumes. Remember that specifying duplexing (backup or migration) also affects recycle output processing.

Note: When you specify either BACKUP or MIGRATION with no subparameter, the default is Y. If you do not specify DUPLEX and either of its subparameters, duplexing does not occur.

For more information about the SETSYS command syntax and the DUPLEX parameter, see *z/OS DFSMShsm Storage Administration*.

Duplex tape status

To display the current duplex status of migration and backup processing, issue the QUERY SETSYS command. Look for message ARC0442I in your output. The example below is the type of information you will receive.

```
ARC0442I TAPE OUTPUT PROMPT FOR TAPECOPY=NO, DUPLEX
ARC0442I (CONT.) BACKUP TAPES=YES, DUPLEX MIGRATION TAPES=(Y,
ARC0442I (CONT.) ERRORALTERNATE=CONTINUE)
```

Functions in progress—that is, migration, backup, or recycle—continue with any duplex values that were first set when the processing began. Therefore, any updates to duplex values become effective only at the start of new processing.

Duplex tape supported functions

Duplex tape supports the following functions:
- Volume backup (including auto-backup)
- Volume migration (including primary space management)
- Recycle
- Backup of migrated data sets
- Backup copy moves from ML1 volumes
- Secondary space management
- Data set migration
- FREEVOL
  - migration volume
  - backup volume
  - ML1BACKUPVERSIONS
- SPILL processing
- ARECOVER ML2 tape
- Data set backup

Note: For the ARECOVER ML2 tape function, only one tape is created. If DUPLEX is specified, DFSMShsm generates internal tape copy requests automatically.

Considerations for duplicating backup tapes

Note: This section refers to backup tapes created by the data set backup by command function. This is not a discussion applicable to all backup tapes.

The TAPECOPY function is an alternative to the DUPLEX tape function.

Using the SETSYS(DSBACKUP) parameter, DEMOUNTDELAY, can cause an output tape to be mounted indefinitely. In this environment, use the duplex tape function when possible. If duplexing cannot be used, use TAPECOPY to make copies of your data set backup tapes.

TAPECOPY of specific backup tapes

If there is a request for TAPECOPY for a specific backup tape and the tape is in use, the TAPECOPY operation fails. The system issues Message ARC0425I indicating the tape is in use. If the tape is in use, use the HOLD BACKUP(DSCOMMAND(SWITCHTAPES)) to demount the tapes in use.

TAPECOPY of nonspecific backup tapes

The TAPECOPY process, when using either ALL or BACKUP keywords, only copies full tapes. If copies of partial backup tapes are desired, use the following process:

- Issue one of the HOLD commands to stop backup for all hosts running backup. The following HOLD commands cause the data set backup tasks that have tapes mounted to demount the tapes at the end of the current data set.
  - HOLD BACKUP holds all backup functions.
  - HOLD BACKUP(DSCOMMAND) holds data set command backup only.
  - HOLD BACKUP(DSCOMMAND(TAPE)) holds data set command backup to tape only.
- A LIST TTOC SELECT(BACKUP NOTFULL) lists the tapes that are not full. This list also shows which tapes do not have an alternate tape.
- A DELVOL MARKFULL is used for those tapes that you do not want extended after the TAPECOPY is made.
- A TAPECOPY OVOL(volser) is used for those tape volsers that do not have an alternate volume indicated in the LIST output.
- When the copies are complete, the corresponding RELEASE BACKUP command issue option can be used (on each host that was held) to allow backup to be usable.

Initial device selection

When MVS selects a device on which to mount a tape, it uses your criteria or its own to determine which device it selects. You can restrict MVS device selection to devices that you specify whether you are in an SMS-managed tape library environment or not. By restricting device selection, you can:

- Ensure that devices with similar characteristics are selected when you want them.
- Ensure that new devices, as they are introduced into the environment, are selected for output.

For more information about device selection, see "Tape device selection" on page 227.

Tape eligibility when output is restricted to a specific nonlibrary device type

If you select an empty tape for output processing, it will always be compatible with the device to which you have restricted output. However, when partially filled tapes are selected, other considerations apply. Table 35 shows which tapes are selected when selection is restricted to a specific device.

Table 35. Initial Tape Selection When Output is Restricted to a Specific Device

Empty	Partial	3480 (No Compaction)	SETSYS NOTAPEHWC 3480X (No Compaction)	SETSYS TAPEHWC 3480X (Compaction)	3490	3590-1
3480		Eligible	Eligible	Eligible	Eligible	Ineligible
	3480	Eligible	Eligible	Ineligible	Ineligible	Ineligible
3480X		Eligible	Eligible	Eligible	Eligible	Ineligible
	3480X (N)	Eligible	Eligible	Ineligible	Ineligible	Ineligible
	3480X (C)	Ineligible	Ineligible	Eligible	Ineligible	Ineligible
3490		Ineligible	Ineligible	Ineligible	Eligible	Ineligible
	3490	Ineligible	Ineligible	Ineligible	Eligible	Ineligible
3590-1		Ineligible	Ineligible	Ineligible	Ineligible	Eligible
	3590-1	Ineligible	Ineligible	Ineligible	Ineligible	Eligible
	3592-J1A	Ineligible	Ineligible	Ineligible	Eligible	Eligible
	3592-E05	Ineligible	Ineligible	Ineligible	Ineligible	Eligible
	3592-E06	Ineligible	Ineligible	Ineligible	Ineligible	Eligible
^	3592-E07	Ineligible	Ineligible	Ineligible	Ineligible	Eligible

Legend:
- N — Noncompacted
- C — Compacted

Note:

1. The IBM 3592 Model J can select only the following media types: MEDIA5, MEDIA6, MEDIA7 and MEDIA8. These tapes are not selectable by any previous technology drives.
2. The IBM 3592 Model E05 and E06 can select only the following media types: MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9 and MEDIA10.
3. The IBM 3592 Model E07 can select only the following media types: MEDIA9, MEDIA10, MEDIA11, MEDIA12, MEDIA13.

Tape eligibility when output is not restricted to a specific nonlibrary device type

When output is not restricted to a specific device, DFSMShsm selects the first available tape. No preference exists between 3490, 3480, 3480X and 3590 devices. The compaction status assigned to an empty 3480X tape is based on a combination of the tape device associated with the tape, and whether either SETSYS TAPEHARDWARECOMPACT or NOTAPEHARDWARECOMPACT has been specified. Table 36 on page 249 shows which tapes are selected when selection is not restricted to a specific group of devices.

Partially full cartridges that are assigned a 3480 device name can be used on a 3480X device, but the data will not be compacted on that tape.

Table 36. Initial Tape Selection When Output is Not Restricted to a Specific Device

Empty	Partial	SETSYS NOTAPEHARDWARECOMPACT	SETSYS TAPEHARDWARECOMPACT
3480		Eligible	Eligible
	3480	Eligible	Eligible
3480X		Eligible	Eligible
	3480X (N)	Eligible	Ineligible
	3480X (C)	Ineligible	Eligible
3490		Eligible	Eligible
	3490	Eligible	Eligible
3590-1		Eligible	Eligible
	3590-1	Eligible	Eligible

Legend:
- N — Noncompacted
- C — Compacted

Note:
1. The IBM 3592 Model J can select only the following media types: MEDIA5, MEDIA6, MEDIA7 and MEDIA8. These tapes are not selectable by any previous technology drives.
2. The IBM 3592 Model E05 and E06 can select only the following media types: MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9 and MEDIA10.
3. The IBM 3592 Model E07 can select only the following media types: MEDIA9, MEDIA10, MEDIA11, MEDIA12, MEDIA13.

Tape eligibility when output is restricted to specific device types

Table 37 (for 3480, 3480X, 3490) and Table 38 on page 250 (for 3590) show which tapes are selected when selection is restricted to specific device types.

Table 37. Tape Eligibility when Output is Restricted to a Specific Device Types (3480, 3480X and 3490). The device restriction is implemented through the ACS routine filtering.

Empty	Partial	3480X (No Compaction)	3480X (Compaction)	3490 (No Compaction), (S)	3490 (Compaction), (S)	3490 (No Compaction), (E)	3490 (Compaction), (E)
3480		Eligible	Eligible	Eligible	Eligible	Ineligible	Ineligible
	3480	Eligible	Ineligible	Ineligible	Ineligible	Ineligible	Ineligible
3480X		Eligible	Eligible	Eligible	Eligible	Ineligible	Ineligible
	3480X (C)	Ineligible	Eligible	Ineligible	Ineligible	Ineligible	Ineligible
	3480X (N)	Eligible	Ineligible	Ineligible	Ineligible	Ineligible	Ineligible
3490 (S)		Eligible	Eligible	Eligible	Eligible	Ineligible	Ineligible
3490 (E)		Ineligible	Ineligible	Ineligible	Ineligible	Eligible	Eligible
	3490 (SC)	Ineligible	Ineligible	Ineligible	Eligible	Ineligible	Ineligible
	3490 (SN)	Ineligible	Ineligible	Eligible	Ineligible	Ineligible	Ineligible
	3490 (EC)	Ineligible	Ineligible	Ineligible	Ineligible	Ineligible	Eligible
	3490 (EN)	Ineligible	Ineligible	Ineligible	Ineligible	Eligible	Ineligible

Table 37. Tape Eligibility when Output is Restricted to a Specific Device Types (3480, 3480X and 3490) (continued). The device restriction is implemented through the ACS routine filtering.

Empty	Partial	3480X (No Compaction)	3480X (Compaction)	3490 (No Compaction), (S)	3490 (Compaction), (S)	3490 (No Compaction), (E)	3490 (Compaction), (E)

Legend:
- S — Standard cartridge system tape
- E — Enhanced cartridge system tape
- N — Noncompacted
- C — Compacted

Table 38 shows which tapes are selected when selection is restricted to the 3590 device type.

Table 38. Tape Eligibility when Output is Restricted to a 3590 Device Type. The device restriction is implemented through the ACS routine filtering.

Empty	Partial	3590 (No Compaction), (S)	3590 (Compaction), (S)	3590 (No Compaction), (E)	3590 (Compaction), (E)
3590 (S)		Eligible	Eligible	Ineligible	Ineligible
3590 (E)		Ineligible	Ineligible	Eligible	Eligible
	3590 (SC)	Ineligible	Eligible	Ineligible	Ineligible
	3590 (SN)	Eligible	Ineligible	Ineligible	Ineligible
	3590 (EC)	Ineligible	Ineligible	Ineligible	Eligible
	3590 (EN)	Ineligible	Ineligible	Eligible	Ineligible

Legend:
- S — Standard cartridge system tape
- E — Enhanced cartridge system tape
- N — Noncompacted
- C — Compacted

Note:
1. The IBM 3592 Model J can select only the following media types: MEDIA5, MEDIA6, MEDIA7 and MEDIA8. These tapes are not selectable by any previous technology drives.
2. The IBM 3592 Model E05 and E06 can select only the following media types: MEDIA5, MEDIA6, MEDIA7, MEDIA8, MEDIA9 and MEDIA10.
3. The IBM 3592 Model E07 can select only the following media types: MEDIA9, MEDIA10, MEDIA11, MEDIA12, MEDIA13.

Allowing DFSMShsm to back up data sets to tape

Note: This section refers to backup tapes created by the data set backup by command function. This is not a discussion applicable to all backup tapes.

Enhancements to data set backup allow the following capabilities:

- Backing up data sets directly to tape:

  Storage administrators define whether tape, ML1 DASD or both are used as target output devices.

  The end user can specify the target output device by using the TARGET keyword for BACKDS, HBACKDS, ARCINBAK, or ARCHBACK. If the storage administrator does not allow the device type specified with the TARGET keyword, the command fails.

If a data set backup command does not specify a target, then the data set will be directed to either tape or DASD based on the size of the data set and the availability of tasks. If the data set is greater than a specified threshold, then the data set will be directed to tape. Any data set less than or equal to a specified threshold will be directed to the first available task, DASD or tape.

- Backup multitasking capability of up to 64 concurrent tasks per host:

  Storage administrators define the number of tasks available to the data set backup function using the SETSYS DSBACKUP command.

  The expansion of data set backup to handle a possible maximum of 64 concurrent tasks, some of which are tape, some of which are DASD, has resulted in the need to balance workload against available and allowed resources.

  Tape takes longer, from initial selection, to first write than DASD, but is potentially faster in data throughput. DASD takes less time to become available since there is no mount delay, but the throughput is potentially less than it is for tapes. For more information, refer to *z/OS DFSMShsm Storage Administration*.
- Concurrent copy enhancements that allow:
  - Users to be notified when the logical concurrent copy is complete.
  - The ability for concurrent copy to override management class attributes for SMS-managed data sets.
  - Concurrent copy capability for SMS and non-SMS data sets.

    Refer to the *z/OS DFSMShsm Storage Administration*, Chapter 6, under the section titled, "Using Concurrent Copy for Data Set Backup" for detailed information.
- Backup volume contention:

  This function allows a backup tape to be taken from a backup or recycle task to satisfy a recover request.

Switching data set backup tapes

Continual mounting of data set backup tapes is possible because the MAXIDLETASKS parameter specifies the tapes be continuously mounted, or because the data set backup workload keeps a tape continuously mounted. Installations can use the SWITCHTAPES function to demount those tapes and deallocate those drives in preparation for disaster backup or the introduction of new tape devices.

The SWITCHTAPES function provides the ability both to plan for the switching of tapes using the DEFINE command or switch tapes as needed for unplanned events by using the HOLD command.

The DEFINE SWITCHTAPES command and its subparameters allow installations to define a time for performance of automatic demounting of data set backup tapes. This automatic demounting of tapes and the deallocation of the tape drives occurs either at a certain time of day or at the end of autobackup. After the tapes are demounted and the drives are deallocated, the data set backup tasks continue with newly selected tapes.

The PARTIALTAPE subparameter of the DEFINE SWITCHTAPES command is used to specify the method that DFSMShsm uses to mark a data set backup output tape as full. PARTIALTAPE(MARKFULL) specifies that all partial tapes demounted with the SWITCHTAPES option are to be marked full. PARTIALTAPE(REUSE) specifies that all partial tapes demounted with the SWITCHTAPES remain in the

DFSMShsm inventory as partial tapes. PARTIALTAPE(SETSYS) specifies use of the SETSYS PARTIALTAPE value. PARTIALTAPE(MARKFULL) is the default.

The HOLD BACKUP(DSCOMMAND(SWITCHTAPES)) option demounts the mounted volumes, and if specified, causes the tapes to be marked full. Any other partial tapes in the DFSMShsm backup inventory remain as selection candidates as partial tapes and may be immediately selected and mounted for output processing. A subsequent RELEASE command is not necessary.

The partial tape status (REUSE or MARKFULL) is taken from the DEFINE SWITCHTAPES command.

Note: If REUSE is in effect during the HOLD BACKUP(DSCOMMAND(SWITCHTAPES)) command, the tapes that were demounted may be selected again for output.

After the tapes are demounted and the drives are deallocated, the data set backup tasks continue with newly selected tapes.

For more information about the specific command syntax and explanations of the commands and parameters affecting the SWITCHTAPES function, see *z/OS DFSMShsm Storage Administration*.

Fast subsequent migration

With fast subsequent migration, data sets recalled from ML2 tape (but not changed, recreated or backed up) can be reconnected to the original ML2 tape. This eliminates unnecessary data movement resulting from remigration and reduces the need to recycle these tapes. Reconnection can occur during individual data set migration or during volume migration. Both SMS and non-SMS data sets are supported; however, reconnection is only supported in a SETSYS USERDATASETSERIALIZATION environment and the fast subsequent migration function will not occur for Hierarchical File System (HFS) data sets.

Note: DFSMShsm performs fast subsequent migration only if the data set has *not* changed since recall. DFSMShsm determines this based on flags in the Format 1 DSCB that are set when the data set is recalled. This allows DFSMShsm to be compatible with other backup applications as DFSMShsm no longer relies on the change bit in the Format 1 DSCB, which may be set or reset by other data set backup products.

Chapter 11. DFSMShsm in a multiple-image environment

At many sites, users must share access to data. Sharing data, however, requires a way to control access to that data.

Example: Users who are updating data need exclusive access to that data; if several users try to update the same data at the same time, the result is a data integrity exposure (the possibility of incorrect or damaged data). In contrast, users who only read data can safely access the same data at the same time.

Note: Multiple DFSMShsm hosts can exist within a single z/OS image, and/or multiple DFSMShsms across multiple z/OS images.

The integrity of its owned and managed data sets is the primary consideration of the DFSMShsm program in a multiple DFSMShsm environment (especially data sets on DASD volumes that two or more systems share). In a single DFSMShsm host environment, DFSMShsm protects the integrity of its owned and managed resources by serializing access to data within a single address space; programs use the ENQ macro to obtain access to a resource and the DEQ macro to free the resource. In a multiple DFSMShsm host environment, DFSMShsm serializes resources by invoking one of the following methods:

- The RESERVE macro to obtain access to a resource and the DEQ macro to free the resource. The RESERVE macro serializes an entire volume against updates made by other z/OS images, but allows shared access between tasks within the same address space, or between address spaces on the owning z/OS image.
- The ENQ macro to obtain access to a resource and the DEQ macro to free the resource. The ENQ macro serializes resources between tasks within the same address space on the same z/OS image. If you need to protect resources between multiple z/OS images, you will need to activate the Global Resource Serialization (GRS) element of z/OS, or a similar product.

  Restriction: The examples used in this publication are based on the GRS element. If you are using another product, consult your products documentation for unique details.
- VSAM record level sharing (RLS) to manage the serialization of the VSAM data sets. RLS enables DFSMShsm to take advantage of the features of the coupling facility.

The trade-offs for each of these serialization methods are discussed in this section as well as other considerations for implementing DFSMShsm in a multiple DFSMShsm host environment.

The following discussions are tasks for you to consider when implementing DFSMShsm in a multiple-image environment:

- “Multiple DFSMShsm host environment configurations” on page 254
- “Defining a multiple DFSMShsm host environment” on page 255
- “Defining a primary DFSMShsm host” on page 255
- “Defining all DFSMShsm hosts in a multiple-host environment” on page 255
- “DFSMShsm system resources and serialization attributes in a multiple DFSMShsm host environment” on page 256
- “Resource serialization in a multiple DFSMShsm host environment” on page 260

Multiple DFSMShsm host environment configurations

Although this information emphasizes the installation of DFSMShsm on only one processor, you may want to install DFSMShsm in a multiple DFSMShsm host configuration after the initial installation.

DFSMShsm can run on z/OS images that are a physical partition, logical partition, or as a guest under VM.

Example of a multiple DFSMShsm host environment

To help demonstrate and clarify the points in this chapter, the following example describes a configuration that can be used for a multiple DFSMShsm host environment.

The XYZ Company has a single processor divided into two separate logical partitions (LPAR). Each LPAR runs an instance of a z/OS image. The first image is the EAST system, and the second the WEST system.

The storage administrators at XYZ Company decide to run four DFSMShsm systems, Host A on the EAST image, and Hosts B, C, and D on the WEST image. All four share a common set of control data sets and journal, and have access to shared DASD containing user data to be managed by DFSMShsm.

There is one MAIN host for each z/OS image. Hosts A and B are MAIN hosts. Hosts C and D are auxiliary (AUX) hosts. Host C is designated as the PRIMARY host, and will perform certain functions on behalf off all hosts to avoid duplicate effort. Figure 74 shows a diagram representing this example.

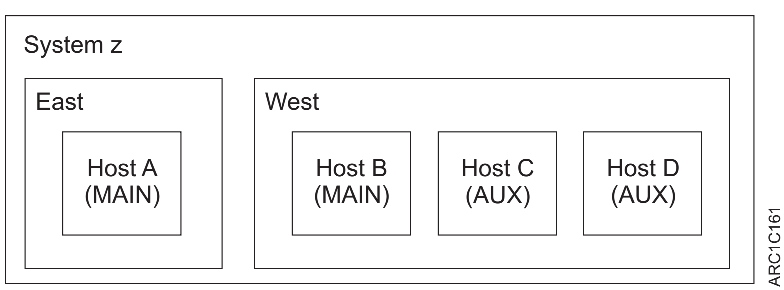


Figure 74. Example Configuration for a Multiple DFSMShsm host Environment

Defining a multiple DFSMShsm host environment

DFSMShsm normally determines that it is in a shared-CDS environment by (1) recognizing startup parameters CDSSHR=YES or RLS, and (2) by examining whether the index component of the migration control data set (MCDS) resides on a DASD volume that has been SYSGENed with a SHARED or SHAREDUP device attribute. If either condition is met, DFSMShsm performs appropriate serialization.

If the control data sets are used by only a single DFSMShsm host but reside on a shared DASD volume, specify CDSSHR=NO to eliminate unnecessary overhead associated with serialization.

If you do not plan to employ RLS, but do plan to share the control data sets using either RESERVE or ENQ methods, you should explicitly specify CDSSHR=YES. This avoids any damage if the DASD volumes are mis-specified as non-shared on one or more of the sharing systems.

Defining a primary DFSMShsm host

In an environment with multiple DFSMShsm hosts (in one or multiple z/OS images), define one host as the “primary DFSMShsm.” This host automatically performs those functions of backup and dump that are not related to one data set or volume. The following “level functions” are included:

- Backing up control data sets as the first phase of automatic backup
- Backing up data sets that have migrated before being backed up
- Moving backup versions of data sets from migration level 1 volumes to backup volumes
- Deleting expired dump copies automatically
- Deleting excess dump VTOC copy data sets

The storage administrator must specify the primary DFSMShsm in the DFSMShsm startup procedure, or as a parameter on the START command. See “Startup procedure keywords” on page 302 for an example of how to specify the primary DFSMShsm with the PRIMARY *keyword*. If no primary DFSMShsm has been specified, DFSMShsm does not perform level functions listed above. If you start more than one primary DFSMShsm, DFSMShsm may process the level functions more than once a day.

The primary host can be either a MAIN or an AUX host. Having an AUX host designated as the primary host reduces contention between its “level functions” and the responsibilities unique to the MAIN host, such as recalls and deletes.

Example: Continuing our example, the XYZ Company makes Host C the designated primary host.

Defining all DFSMShsm hosts in a multiple-host environment

In a multiple DFSMShsm host environment, ensure that the host identifier for each host is unique by considering how you specify the HOST=*x* keyword of the DFSMShsm startup procedure. *x* represents the unique host identifier for each host. For details about specifying the HOST=*x* keyword, see “HOST=x” on page 303.

If you choose to use a single startup procedure in starting multiple DFSMShsm hosts in a single z/OS image, you have two alternatives to identify these startups for subsequent MODIFY commands:

```
S procname.id1,HOST=B,HOSTMODE=MAIN,other parms
S procname.id2,HOST=C,PRIMARY=YES,HOSTMODE=AUX,other parms
```

(The common procedure should specify PRIMARY=NO, so that you only have to override it only for the one primary host.)

or

```
S procname,JOBNAME=id1,HOST=B,HOSTMODE=MAIN,other parms
S procname,JOBNAME=id2,HOST=C,PRIMARY=YES, HOSTMODE=AUX,other parms
```

If you need to issue the same command to multiple DFSMShsm hosts started with identifiers that have the same set of leading characters, you can use an asterisk wildcard with the MODIFY command:

```
F id*,command
```

DFSMShsm system resources and serialization attributes in a multiple DFSMShsm host environment

Table 39 on page 257 shows DFSMShsm-related global serialization resources, descriptions of their purposes, and considerations or restrictions on how they are protected.

Global Resource Serialization (GRS) supports two topologies: GRS Ring and GRS Star. A GRS Star requires a coupling facility connected to all hosts in a parallel sysplex. A GRS Ring does not exploit the coupling facility, but requires a common sysplex timer connected to the hosts in or outside of the parallel sysplex. The collection of z/OS images connected in either a GRS Ring or GRS Star topology is termed GRSplex.

The table is divided into four subsections: serialization of control data sets, serialization of functions, serialization of user data sets by DFSMShsm, and serialization of user data sets by the system.

In the table, each pair of **QNAME** and **RNAME** values uniquely identifies a resource; the **SCOPE** indicates the range of control. The **DISP** (disposition) column indicates whether DFSMShsm or the system requests the resource for exclusive (EXCL) or shared (SHR) use. The significance of the **CAN CONVERT FROM RESERVE** column is explained in "Converting from volume reserves to global resource serialization" on page 265. The **MUST PROPAGATE** column indicates whether GRS must communicate the ENQ to all shared systems.

If you use Global Resource Serialization, you can use this table for informational purposes to better understand global serialization. However, if you use a product other than but similar to GRS, you should study this table carefully to determine if you must adapt that product to take particular actions.

Note: The RNAMEDSN parameter may affect the minor names shown in the following table. For a discussion of the RNAMEDSN parameter and GRSplex serialization, see Chapter 12, “DFSMShsm in a sysplex environment,” on page 279.

Table 39. DFSMShsm-Related Global Serialization Resources

QNAME (MAJOR)	RNAME (MINOR)	SCOPE	DISP	CAN CONVERT FROM RESERVE	MUST PROPAGATE	DESCRIPTION
SERIALIZATION OF CONTROL DATA SETS						
ARCAUDIT 1	ARCBCDS ARCMCDS ARCOCDS	SYSTEMS	SHR	NO	N.A. 2	The associated volume reserves of the CDS volumes prevent updates from other processors while AUDIT FIX is running in the processor issuing the reserve.
ARCBACK 1	ARCBCDS ARCMCDS ARCOCDS	SYSTEMS	SHR	NO	N.A.	The associated volume reserves of the CDS volumes prevent access or updates to the control data sets from other processors while the control data sets are being backed up in the processor issuing the reserve.
ARCGPA 1	ARCBCDS ARCMCDS ARCOCDS	SYSTEMS	SHR	NO	N.A.	The associated volume reserves of the CDS volumes prevent access or updates to the control data sets from other processors while CDS updates are being made.
ARCGPA	ARCRJRN	SYSTEMS	EXCL	YES	YES	The associated volume reserve of the journal volume prevents access to the journal from other processors while the journal is being read or written in the processor issuing the reserve.
ARCGPAL 1	ARCMCDS	SYSTEMS	SHR	NO	N.A.	The associated volume reserve of the MCDS volume prevents access to the level 2 control record (L2CR) from other processors while the L2CR is being updated in the processor issuing the reserve.
ARCUPDT 1	ARCBCDS ARCMCDS ARCOCDS	SYSTEMS	SHR	NO	N.A.	The associated volume reserves of the CDS volumes prevent access to the control data sets from other processors while they are being recovered by UPDATEC processing in the processor issuing the reserve.
ARCENQG 3	ARCBCDS ARCMCDS ARCOCDS	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued in CDSQ=YES environments and is held while accessing the control data sets from the processor issuing the enqueue.
ARCENQG	ARCCDSVF	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued to ensure that only one CDS version backup function is running in the HSMplex.
ARCENQG	ARCCDSVD	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued to ensure that the data area used in CDS version backup is not updated by any sharing processor while the function is running.

Table 39. DFSMShsm-Related Global Serialization Resources (continued)

QNAME (MAJOR)	RNAME (MINOR)	SCOPE	DISP	CAN CONVERT FROM RESERVE	MUST PROPAGATE	DESCRIPTION
ARCENQG (RLS mode)	ARCCAT	SYSTEMS	SHR	N.A.	YES	This enqueue is issued as a CDS update resource only in RLS mode.
ARCENQG (RLS mode)	ARCCAT	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued as a CDS backup resource only in RLS mode.
SERIALIZATION OF FUNCTIONS						
ARCENQG	ARCBMBC	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued in SETSYS USERSERIALIZATION environments and ensures that only one instance of move backup versions is running in the HSMplex.
ARCENQG	ARCL1L2	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued in SETSYS USERSERIALIZATION environments and ensures that only one instance of level 1 to level 2 migration is running in the HSMplex.
ARCENQG	ARCMCLN	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued in SETSYS USERSERIALIZATION environments and ensures that only one instance of migration cleanup is running in the HSMplex.
ARCENQG	EXPIREBV	SYSTEMS	EXCL	N.A	YES	There is a limit of one instance of the EXPIREBV command in an HSMplex.
ARCENQG	RECYC-L2	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued during RECYCLE to establish a limit of one instance of recycling ML2 tapes in the HSMplex.
ARCENQG	RECYC-DA	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued during RECYCLE to establish a limit of one instance of recycling daily backup tapes in the HSMplex.
ARCENQG	RECYC-SP	SYSTEMS	EXCL	N.A.	YES	This enqueue is issued during RECYCLE to establish a limit of one instance of recycling spill backup tapes in the HSMplex.
ARCENQG	COPYPOOL|| *cpname*	SYSTEMS	EXCL	N.A.	YES	The scope of the fast replication copy pool extends beyond an HSMplex because a copy pool is defined at the SMSplex level. All DFSMShsm hosts, regardless of which HSMplex they reside in, are prevented from processing the same copy pool. The resource is obtained unconditionally and if the resource is not immediately available, it waits.
ARCENQG	CPDUMP&& *cpname*&&V*nnn*	SYSTEMS	EXCL	N.A.	YES	This enqueue is used for the dumping of copy pools.
ARCBTAPE	volser	SYSTEMS	EXCL	N.A.	YES	This enqueue is used for the Recover Tape Takeaway function.

Table 39. DFSMShsm-Related Global Serialization Resources (continued)

QNAME (MAJOR)	RNAME (MINOR)	SCOPE	DISP	CAN CONVERT FROM RESERVE	MUST PROPAGATE	DESCRIPTION
ARCBTAPE	volser. TAKEAWAY	SYSTEMS	EXCL	N.A.	YES	This enqueue is used for the Recover Tape Takeaway function.
SERIALIZATION OF USER DATA SETS BY DFSMShsm						
ARCDSN	*dsname*	SYSTEMS	4	N.A.	YES	For SETSYS USERSERIALIZATION environments, this enqueue enables DFSMShsm to protect the integrity of the data set related to concurrent processing in all HSMplexes. For SETSYS HSERIALIZATION environments, this enqueue enables DFSMShsm to protect the integrity of the data set related to concurrent processing only within the DFSMShsm processor issuing the enqueue.
ARCENQG	*dsname*	SYSTEMS	5	N.A.	YES	This ENQ prevents catalog locate requests from getting a "not cataloged" response in that interval of time during migration or recall when the volser is being changed from MIGRAT to non-MIGRAT or from non-MIGRAT to MIGRAT. It is also used to determine whether a recall that has an "in process" flag set on really means "in process" or is a residual condition after a system outage.
ARCBACV	*volserx* 6	SYSTEMS	EXCL	YES	YES	This reserve is issued only when running in a SETSYS HSERIALIZATION environment when doing volume backup. The associated volume reserve of the user volume prevents updates of a user data set from other processors while it is being copied by the processor issuing the reserve.
ARCMIGV	*volserx* 6	SYSTEMS	EXCL	YES	YES	This reserve is issued only when running in a SETSYS HSERIALIZATION environment when doing volume migration. The associated volume reserve of the user volume prevents updates of a user data set from other processors while it is being copied by the processor issuing the reserve.
SERIALIZATION OF USER DATA SETS BY THE SYSTEM						
SYSDSN	*dsname*	7	5	N.A.	YES	This enqueue is the method MVS allocation uses to provide data integrity when allocating data sets.

Table 39. DFSMShsm-Related Global Serialization Resources (continued)

QNAME (MAJOR)	RNAME (MINOR)	SCOPE	DISP	CAN CONVERT FROM RESERVE	MUST PROPAGATE	DESCRIPTION
SYSVSAM	*dsname*	SYSTEMS	5	N.A.	YES	This enqueue is the method VSAM uses to provide data integrity commensurate with the share options of VSAM data sets.
SYSVTOC	*volser*	SYSTEMS	5	YES	YES	This enqueue is the method DFSMSdfp DADSM uses to provide integrity of a volume's VTOC. **Note:** For more information about journal volume dumps, see "DFSMSdss Considerations for dumping the journal volume" on page 268.

1 These resources are requested only in CDSR=YES environments. In CDSQ=YES environments, serialization is achieved via a global enqueue using the QNAME of ARCENQG.

2 N.A. means Not applicable

3 Valid if CDSQ = YES

4 EXCL is used for migration, recall, recover, and ARECOVER. SHR is used for backup and ABACKUP.

5 The DISP of this resource can be either EXCL or SHR. There are some requests of each type.

6 Used only in SETSYS DFHSMDATASETSERIALIZATION

7 The SCOPE of the SYSDSN resource is SYSTEM only but is automatically propagated by GRS, if the default GRSRNL00 member supplied by MVS is being used. GRS-equivalent products also need to propagate this resource.

Resource serialization in a multiple DFSMShsm host environment

Data set integrity is of major importance. In both a single DFSMShsm host environment and a multiple DFSMShsm host environment, serialization of resources ensures their integrity. DFSMShsm serializes data sets with either of two methods:

Volume reserve
: DFSMShsm issues reserves against the source volume to protect data sets during volume processing. The protection is requested by the z/OS RESERVE macro.

Global enqueue
: User data sets can be protected with the z/OS ENQ and z/OS DEQ macros.

For information on resource serialization in an HSMplex, see "Resource serialization in an HSMplex environment" on page 280.

Global resource serialization

Global resource serialization (GRS) is a z/OS element designed to protect the integrity of resources in a multiple DFSMShsm host environment. By combining the systems that access the shared resources into a global resource serialization complex and by connecting the systems in the GRS complex with dedicated communication links, GRS serializes access to shared resources. User data sets are protected by associating them with the SYSDSN resource and then passing the SYSDSN token to the other images in the GRSplex.

The SYSDSN resource, provided by using the ENQ and DEQ macros, is passed to cross-system (global) enqueues. DFSMShsm shares its resources according to z/OS-defined ranges of control known as *scopes* in GRS terminology.

In a multiple DFSMShsm host environment where GRS is used for CDS serialization, functional processing, and some data set level serialization, the integrity of DFSMShsm resources is dependent on the QNAME=ARCENQG generic resource name being propagated to all systems.

In a sysplex environment, a GRSplex is one or more z/OS systems that use global serialization to serialize access to shared resources. You can now place multiple HSMplexes (one or more installed processors that share common MCDS, BCDS, OCDS, and journals) in the same GRSplex, because DFSMShsm can now translate minor global resource names to avoid interference between HSMplexes. For more information on global serialization in a sysplex, see Chapter 12, “DFSMShsm in a sysplex environment,” on page 279.

These scopes, in association with GRS resource name lists (RNLs), define to the entire complex which resources are local and which resources are global. The GRS scopes are:

STEP Scope within a z/OS address space

SYSTEM
Scope within a single z/OS image

SYSTEMS
Scope across multiple z/OS images

The GRS resource name lists (RNLs) are:

SYSTEM inclusion RNL
Lists resources requested with a scope of SYSTEM that you want GRS to treat as global resources.

SYSTEMS exclusion RNL
Lists resources requested with a scope of SYSTEMS that you want GRS to treat as local resources.

RESERVE conversion RNL
Lists resources requested on RESERVE macro instructions for which you want GRS to suppress the hardware reserve.

In a GRS environment, some available DFSMShsm and JES3 options can affect the overall performance of DFSMShsm. For more information about exclusion and conversion RNLs, refer to *z/OS MVS Planning: Global Resource Serialization*.

Serialization of user data sets

The basis of DFSMShsm serialization for non-VSAM user data set processing on a single DFSMShsm host environment is the SYSDSN resource. Serialization of VSAM user data sets use both SYSDSN and SYSVSAM resources. DFSMShsm can explicitly request the SYSDSN resource or can obtain the SYSDSN resource via allocation of the data set. The SYSVSAM resource can be requested only during the open of the data set. Table 40 on page 262 describes the serialization for user data sets as DFSMShsm processes them.

Table 40. Single DFSMShsm host Environment User Data Set Serialization

Function	Data Set Organization	Share Options	Serialization
Migration	All	—	Exclusive
Backup	VSAM	—	Exclusive **Note:** DFSMShsm does no explicit synchronization, but DFSMSdss does a shared enqueue on SYSDSN and an exclusive enqueue on SYSVSAM.
	Non-VSAM	—	Shared

Exclusive control results in the correct level of control possible in the systems environment.

In a single DFSMShsm host environment, the preceding serialization is all that is needed. In a multiple DFSMShsm host environment, update protection must be extended to the other processors. That protection can be achieved with GRS or an equivalent cross-system enqueue product.

Serialization of control data sets

The method with which DFSMShsm serializes control data sets depends on whether GRS is installed.

Serialization of control data sets with global resource serialization

When DFSMShsm is running in a multiple DFSMShsm host environment with GRS (or a similar global enqueue product) installed, the method that DFSMShsm uses to serialize its control data sets depends on the CDSQ and CDSR keywords specified in the DFSMShsm startup procedure.

These keywords direct DFSMShsm to enable global enqueue serialization with GRS, to serialize the control data sets by reserving the volumes on which the control data sets reside, or to serialize the control data sets by VSAM RLS.

CDSQ keyword of the DFSMShsm startup rrocedure: When you specify CDSQ=YES, DFSMShsm serializes the control data sets (between multiple DFSMShsm hosts) with a global (SYSTEMS) exclusive enqueue while allowing multiple tasks within a single DFSMShsm host environment to access the control data sets concurrently. This optional CDS serialization technique is implemented with a SCOPE=SYSTEMS resource that enables GRS or a similar product to enqueue globally on the resource as an alternative to reserving hardware volumes. All hosts in an HSMplex must implement the same serialization technique and must propagate the QNAME of ARCENQG as shown in Table 42 on page 264. Do not specify CDSQ=YES in the DFSMShsm startup procedure unless you have a cross-system serialization product propagating the ARCENQG resource.

To use the AUX mode of DFSMShsm, you must specify CDSQ=YES or CDSSHR=RLS in each startup procedure. If the HSMplex consists of an AUX mode host and more than one z/OS image then you must use GRS, or a similar product to propagate enqueues.

CDSR keyword of the DFSMShsm startup procedure: When you specify CDSR=YES and CDSQ=NO, DFSMShsm serializes the control data sets with a shared ENQ/RESERVE. This means that all DFSMShsm hosts in your HSMplex must have HOSTMODE=MAIN and must implement the same serialization technique.

When the serialization technique has not been specified, the default serialization technique depends on the value of HOSTMODE:
- If HOSTMODE=MAIN, DFSMShsm assumes CDSR=YES.
- If HOSTMODE=AUX, DFSMShsm indicates an error with message ARC0006I.

When using CDSR=YES, if two or more DFSMShsm hosts are started simultaneously, small windows exist where lockouts can occur. You may want to consider using CDSQ instead.

CDSSHR keyword of the DFSMShsm startup procedure: When you specify CDSSHR=YES, DFSMShsm serializes the control data sets with the type of multiprocessor serialization requested by the CDSQ and CDSR keywords. However, if you specify CDSSHR=RLS, DFSMShsm performs multiprocessor serialization using RLS. Specifying CDSSHR=NO performs no multiple DFSMShsm host environment serialization at all.

Table 41 shows the serialization techniques available with varying combinations of the startup procedure keywords.

Table 41. DFSMShsm Serialization with Startup Procedure Keywords

CDSQ Keyword	CDSR Keyword	CDSSHR Keyword	Serialization
YES	YES	YES	Both CDSQ and CDSR options are used.
YES	NO or not specified	YES	Only the CDSQ option is used.
With any other combination of specifications		YES	Only the CDSR option is used.
--	--	RLS	Uses VSAM RLS
--	--	NO	No multiprocessor serialization. No other processor shared control data sets.

When the CDSQ and CDSR keywords are specified, DFSMShsm monitors each processor's updates to the control data sets and ensures that the serialization technique of the processor making the current update is identical to the serialization technique of the process that has made the previous update.

"Startup procedure keywords" on page 302 describes the keywords for the DFSMShsm startup procedure. Additionally, an example of the DFSMShsm startup procedure can be found in topic "Starter set example" on page 109.

Table 42 on page 264 shows the resource names for the control data sets when they are protected with GRS.

Table 42. DFSMShsm Resource Names for Control Data Sets

Major (qname) resource name	Minor (rname) resource name	Serialization result
ARCENQG	ARCMCDS	This enqueue allows global resource serialization of the DFSMShsm MCDS.
	ARCBCDS	This enqueue allows global resource serialization of the DFSMShsm BCDS.
	ARCOCDS	This enqueue allows global resource serialization of the DFSMShsm OCDS.
ARCGPA	ARCRJRN	This enqueue allows only one processor to backup the journal.

Serialization of control data sets without global resource serialization

When DFSMShsm is running in a z/OS image environment without a global enqueue product, DFSMShsm serializes CDS processing by issuing a volume reserve against the volume on which the CDS resides. Because DFSMShsm reserves the volume it is processing, other z/OS images cannot update the control data sets while they are being processed by DFSMShsm.

Serialization of DFSMShsm functional processing

Serialization is required to ensure that only one processor at a time can process DFSMShsm-owned data. There is no cross-system protection without GRS-type processing.

Table 43 describes the resource names for DFSMShsm processing when it is protected by GRS.

Table 43. DFSMShsm Resource Names

Major (qname) Resource Name	Minor (rname) Resource Name	Serialization result
ARCENQG	ARCL1L2	This enqueue allows only one DFSMShsm host to perform level 1 to level 2 migration.
	ARCMCLN	This enqueue allows only one DFSMShsm host to perform migration cleanup.
	ARCBMBC	This enqueue allows only one DFSMShsm host to move backup versions.
	RECYC-L2	This enqueue allows only one DFSMShsm host to perform recycle on ML2 tape volumes.
	RECYC-SP	This enqueue allows only one DFSMShsm host to perform recycle on spill tape volumes.
	RECYC-DA	This enqueue allows only one DFSMShsm host to perform recycle on daily tape volumes.
	EXPIREBV	There is a limit of one instance of the EXPIREBV command in an HSMplex.

Choosing a serialization method for user data sets

The SETSYS DFHSMDATASETSERIALIZATION | USERDATASETSERIALIZATION command determines whether user data sets are serialized with volume reserves or global enqueues. DFHSMDATASETSERIALIZATION is the default.

DFHSMDATASETSERIALIZATION

The SETSYS DFHSMDATASETSERIALIZATION option directs DFSMShsm to serialize user data sets during volume migration and volume backup processing.

If you specify . . .	Then . . .
SETSYS DFHSMDATASETSERIALIZATION	DFSMShsm serializes data sets processed by volume migration and volume backup by reserving the source volume during data set processing. DFSMShsm releases the volume after the data copy, as each volume's data sets migrate or are backed up.

Performance considerations

Users who specify the SETSYS DFSMSHSMDATASETSERIALIZATION option will not receive the performance improvement to the incremental backup function that is introduced in DFSMShsm Version 1 Release 5 or use Fast Subsequent Migration introduced in Release 10. Only use the SETSYS DFHSMDATASETSERIALIZATION command if your environment requires it. Otherwise, use the SETSYS USERDATASETSERIALIZATION command.

Volume reserve considerations

Although volume reserves ensure data set integrity, they also prevent users on other systems from accessing other data sets on the reserved volume. In addition, if one processor issues multiple reserves for the same device, that processor can tie up a device. Other processors cannot access the shared device until the reserve count is zero and the reserving processor releases the shared device.

USERDATASETSERIALIZATION

The SETSYS USERDATASETSERIALIZATION option indicates either that other processors do not share volumes or that a product such as GRS or JES3 provides global data set serialization.

If you specify . . .	Then . . .
SETSYS USERDATASETSERIALIZATION	DFSMShsm serializes data sets processed by volume migration and volume backup by serializing (ENQ) only the data set (and not the volume) during data set processing. **Note:** To prevent the possibility of a deadlock occurring with volume reserves, any multivolume, physical sequential, SMS-managed data sets are supported only when the SETSYS USERDATASETSERIALIZATION command has been specified. For more information about these parameters, see *z/OS DFSMShsm Storage Administration*.

Converting from volume reserves to global resource serialization

If you are using volume reserves, you can convert selected DFSMShsm volume reserves to global enqueues with GRS. You identify resources by specifying resource names (RNAMES) in a GRS resource name list (RNL). The resource names are organized into groupings of one QNAME and one or more associated RNAMEs. The RNAMEs are grouped into exclusion lists or conversion lists according to the priority of other programs sharing the volume with

DFSMShsm-processed data sets. Figure 75 and Figure 77 on page 268 show the resource names for which DFSMShsm issues reserves.

Some resources must not be converted from volume reserves. Figure 75 and Figure 77 on page 268 each show two tables.

Setting up the GRS resource name lists

Planning your RNLs is key to implementing a GRS strategy. The RNLs are lists of resource names, each with a QNAME (major name) and one or more associated RNAMES (minor name).

Example: DFSMShsm serialization configuration

Figure 75 illustrates the expected configuration of DFSMShsm resources when the CDSR=YES and DFHSMDATASETSERIALIZATION options are being used. The control data sets and the journal are in the SYSTEMS exclusion RNL with a QNAME of ARCGPA and RNAMES of ARCMCDS, ARCBCDS, ARCOCDS, and ARCRJRN. The exclude of the ARCGPA resource allows the reserves to protect the control data sets. Therefore, CDS backup and journal backup are serialized by volume reserves and not by global enqueue serialization.

The RESERVE conversion RNL entries, ARCBACV and ARCMIGV, are meaningful only if the SETSYS DFHSMDATASETSERIALIZATION command has been specified. If SETSYS USERDATASETSERIALIZATION has been specified, reserves using these RNAMES are not issued.

SYSTEMS Exclusion Resource Name List

Resources that MUST NOT be converted from volume reserves

QNAME	RNAMES
* ARCAUDIT	ARCMCDS ARCBCDS ARCOCDS
* ARCBACK	ARCMCDS ARCBCDS ARCOCDS
* ARCGPA	ARCMCDS ARCBCDS ARCOCDS ARCRJRN
* ARCGPAL	ARCMCDS
* ARCUPDT	ARCMCDS ARCBCDS ARCOCDS

RESERVE Conversion Resource Name List

Resources that may be converted to global enqueues

QNAME	RNAMES
ARCBACV	volser1,...volserx
ARCMIGV	volser1,...volserx

* = shared resource

ARC1CI01

Figure 75. Access Priority Name List (Configuration 1)

Note: ARCBACV and ARCMIGV should be converted only if GRS or a GRS-like product propagates enqueues for the SYSDSN resource to all shared systems. Converting these reserves without this cross-system propagation of SYSDSN enqueues removes the necessary cross-system serialization and risks loss of data.

Figure 76 on page 267 is an example of the GRS RNLDEF statements for Figure 75.

```
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCAUDIT)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCBACK)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCGPA)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCGPAL)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCUPDT)
RNLDEF RNL(CON) TYPE(GENERIC) QNAME(ARCBACV)
RNLDEF RNL(CON) TYPE(GENERIC) QNAME(ARCMIGV)
```

Figure 76. RNLDEF Statements that Define the Example Configuration

Alternate example: DFSMShsm serialization configuration

Now assume that you must place other data on the volume with the journal. Because the volume has other data set activity, you choose to protect the journal with global enqueue serialization instead of volume reserves because global enqueue serialization serializes at the data set level. This allows us to concurrently access the journal as well as any other data sets on the volume. However, while this illustrates the conversion from volume reserves to global enqueue serialization, this implementation is neither recommended or likely to be justified because the journal data set is the single most active data set whenever DFSMShsm is running and the preferred implementation is to place the journal on its own volume. Indicate that you want the journal resource, ARCRJRN, converted from a volume reserve to a global enqueue by placing the journal resource in the RESERVE conversion RNL.

Figure 77 on page 268 is identical to Figure 75 on page 266, except that you have moved the journal into the RESERVE conversion RNL.

The journal resource can be adequately protected by either a global enqueue or a reserve so ensure that the journal is placed in either the exclusion list or the conversion list. If the journal does not appear in either of the lists, DFSMShsm serializes the resource with both a hardware reserve and a global enqueue causing an unnecessary performance degradation.

The RESERVE conversion RNL entries, ARCBACV and ARCMIGV, are meaningful only if the SETSYS DFHSMDATASETSERIALIZATION command has been specified. If SETSYS DFHSMDATASETSERIALIZATION has not been specified, reserves using these resource names are not issued.

SYSTEMS Exclusion Resource Name List

Resources that MUST NOT be converted from volume reserves

QNAME	RNAMES
* ARCAUDIT	ARCMCDS ARCBCDS ARCOCDS
* ARCBACK	ARCMCDS ARCBCDS ARCOCDS
* ARCGPA	ARCMCDS ARCBCDS ARCOCDS
* ARCGPAL	ARCMCDS
* ARCUPDT	ARCMCDS ARCBCDS ARCOCDS

RESERVE Conversion Resource Name List

Resources that may be converted to global enqueues

QNAME	RNAMES
ARCBACV	volser1,...volserx
ARCMIGV	volser1,...volserx
ARCGPA	ARCRJRN

* = shared resource

ARC1CI02

Figure 77. Access Priority Name List (Configuration 2)

Note: ARCBACV and ARCMIGV should be converted only if GRS or a GRS-like product propagates enqueues for the SYSDSN resource to all shared systems. Converting these reserves without this cross-system propagation of SYSDSN enqueues removes the necessary cross-system serialization and risks loss of data.

Figure 78 is an example of the GRS RNLDEF statements for Figure 77.

```
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCAUDIT)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCBACK)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(ARCGPA) RNAME('ARCMCDS ')
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(ARCGPA) RNAME('ARCBCDS ')
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(ARCGPA) RNAME('ARCOCDS ')
RNLDEF RNL(CON) TYPE(SPECIFIC) QNAME(ARCGPA) RNAME('ARCRJRN ')
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCGPAL)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCUPDT)
RNLDEF RNL(CON) TYPE(GENERIC) QNAME(ARCBACV)
RNLDEF RNL(CON) TYPE(GENERIC) QNAME(ARCMIGV)
```

Figure 78. RNLDEF Statements That Define the Alternate Configuration

DFSMSdss Considerations for dumping the journal volume

If the volume containing DFSMShsm's journal data set is dumped by invoking DFSMSdss for a full volume dump, then the DFSMShsm journal resource

```
QNAME=ARCGPA, RNAME=ARCRJRN
```

and the SYSTEM resource

```
QNAME=SYSVTOC, RNAME=volser-containing-journal
```

must be treated consistently, that is, both treated as local resources or both treated as global resources.

Attention: Failure to treat journal resources consistently may result in lockouts or long blockages of DFSMShsm processing.

As most customers treat the SYSVTOC resource generically, you will most likely serialize the journal resource the same way you do the SYSVTOC. **Example:** If you exclude the SYSVTOC, you will also exclude the journal by using the following statements:

```
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVTOC)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(ARCGPA)
```

or if you convert the SYSVTOC, you can convert the journal by using the following statements:

```
RNLDEF RNL(CON) TYPE(GENERIC) QNAME(SYSVTOC)
RNLDEF RNL (EXCL) TYPE (SPECIFI) QNAME (ARCGPA) RNAME('ARCMCDS ')
RNLDEF RNL (EXCL) TYPE (SPECIFI) QNAME (ARCGPA) RNAME('ARCBCDS ')
RNLDEF RNL (EXCL) TYPE (SPECIFI) QNAME (ARCGPA) RNAME('ARCOCDS ')
RNLDEF RNL(CON) TYPE(SPECIFIC) QNAME(ARCGPA) RNAME('ARCRJRN ')
```

DFSMShsm data sets in a multiple DFSMShsm host environment

This section discusses considerations for DFSMShsm data sets in a multiple DFSMShsm host environment. Chapter 3, "DFSMShsm data sets," on page 9 describes the data sets DFSMShsm requires for full-function processing. These important data sets, used by DFSMShsm for control, record keeping, reporting, and problem analysis, are the very heart of DFSMShsm.

CDS considerations in a multiple DFSMShsm host environment

In a multiple DFSMShsm host environment, the control data sets must meet the following conditions:

- They must reside on shared DASD. If the DFSMShsm startup procedure does not detect that the volume containing the MCDS is allocated as a shared volume or if CDSSHR=RLS is not specified, DFSMShsm does not do multihost serialization (global enqueues or volume reserves) when it accesses user data sets. If the unit control block of the volume containing the MCDS index is marked as shared, DFSMShsm performs what it calls "multiple processor serialization". Implementing CDSQ-only serialization in a multiple z/OS image environment requires that GRS propagate the enqueues to other z/OS images.
- You should specify your desired type of CDS serialization with the CDSQ, CDSR, or CDSSHR keywords described in "Serialization of control data sets" on page 262. If you specify CDSQ=YES, your control data sets are associated with a major resource name of ARCENQG and a minor resource name of ARC*x*CDS. Implementing CDSQ serialization requires that GRS or a similar enqueue product does a global enqueue of these resources.

  If you specify CDSR=YES, your control data sets are associated with a major resource name of ARCGPA and a minor resource name of ARC*x*CDS. GRS or a similar enqueue product is *not* required to implement a CDSR serialization. Be certain not to convert the reserve.

  If you specify CDSSHR=RLS, your control data sets are accessed in record level sharing (RLS) mode. CDSSHR=RLS ignores the CDSQ and CDSR options. GRS or a similar enqueue product is required to implement RLS serialization.

Preventing interlock of DFSMShsm control data sets

In a multiple DFSMShsm host environment, you must provide protection to prevent control data sets from entering into interlock (deadlock) situations.

VSAM SHAREOPTIONS parameters for control data sets

There are two methods for defining the share options for DFSMShsm CDSs. The first method is for starting one DFSMShsm host and the second method is for starting more than one DFSMShsm host.

Method 1—VSAM SHAREOPTIONS(2 3): If you are starting only one DFSMShsm host in a z/OS image, the following share option strategy provides maximum protection against accidental, non-DFSMShsm concurrent updates:

- Define the CDSs with VSAM SHAREOPTIONS(2 3).
- Use the GRS RNL exclusion capability to avoid propagating the VSAM resource of SYSVSAM for the CDS components to other systems.

This share option can also be used with RLS when starting DFSMShsm in a multiple DFSMShsm host environment under a single z/OS image.

Note: The GRS RNLDEF statements cannot be used with method 1 when using RLS.

Cross-region share option 2 allows only one processor at a time to open a data set for output. If that data set is in the SYSTEMS exclusion list, the open is limited to a single z/OS system. This combination sets a limit of one open per processor with the expectation that the one open will be DFSMShsm. As long as DFSMShsm is active in each z/OS ystem, then no jobs, including authorized jobs, can update the CDSs.

If you define the CDSs with VSAM SHAREOPTIONS(2 3) and start DFSMShsm on multiple host environments, exclude the SYSVSAM resource related to the CDS components from being passed around the GRS ring. Figure 79 shows the RNLDEF statements that exclude the SYSVSAM resource from being passed around the GRS ring.

```
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(MCDS index name)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(MCDS data name)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(BCDS index name)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(BCDS data name)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(OCDS index name)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSVSAM) RNAME(OCDS data name)
```

Figure 79. GRS RNLDEF Statements for SHAREOPTIONS(2 3)

Attention:

1. Reserve contentions can occur when a site does *not* use a global serialization product and that site processes DFSMShsm and applications concurrently with VSAM data sets on the same volume.
2. Specify DISP=SHR for read-only utilities (**Example:** DCOLLECT). Specify DISP=OLD (which allows an exclusive enqueue on the data set) for utilities that alter the CDSs (**Example:** AMS REPRO and IMPORT).
3. Do not attempt to reorganize your CDSs while DFSMShsm is running on any processor that uses those CDSs. See “HSMPRESS” on page 147 for a sample job for reorganizing CDSs.

4. You can use this share option to access the CDSs in RLS mode. When you use RLS in a VSAM SHAREOPTIONS(2 3) mode, VSAM allows non-RLS access that is limited to read processing. **Example:** EXAMINE or REPRO access.

Method 2—VSAM SHAREOPTIONS(3 3): The starter set defines the MCDS, BCDS, and OCDS with VSAM SHAREOPTIONS(3 3) to allow DFSMShsm to be started in a multiple DFSMShsm host environment with GRS or equivalent function. If you are starting more than one DFSMShsm host in an z/OS image, you can use this share-option strategy. DFSMShsm provides an appropriate multiple processor serialization protocol to ensure read and update integrity of the CDSs accessed by multiple DFSMShsm processors. You can use this share option to access the CDSs in RLS mode.

Considerations for VSAM SHAREOPTIONS:

With either of the preceding VSAM SHAREOPTIONS, a data integrity exposure could still exist if DFSMShsm is not active in all connected processors. Therefore, strictly controlled procedures need to be in place for doing periodic maintenance on the CDSs. **Example:** If either of the VSAM SHAREOPTIONS has been defined and the CDSs are on shared DASD, a DFSMShsm system can be started on one processor while a utility job is reorganizing the CDSs on another processor. Problems can be avoided by allocating the utility job with a disposition of OLD, which causes an exclusive enqueue on the SYSDSN resource for the CDS cluster name. Because DFSMShsm must have a shared enqueue on the same resource, this approach prevents DFSMShsm from running at the same time as the utility in a GRS environment. See “HSMPRESS” on page 147 for a sample listing of a utility job that allocates with the disposition of OLD.

Before any I/O processing begins, DFSMShsm reserves the volume containing the CDSs. To prevent changes to the CDSs, DFSMShsm reserves these volumes for a comparatively long time during the following processes:

- Automatic backup of the MCDS, BCDS, and OCDS
- BACKVOL CONTROLDATASETS command
- AUDIT FIX command
- UPDATEC command

After processing has been completed, DFSMShsm releases the volumes.

CDS backup version considerations in a multiple DFSMShsm host environment

In a multiple DFSMShsm host environment, the way you set up your CDS backup version copies depends on the devices on which you back up your CDSs.

DASD CDS backup versions

If you are backing up your CDSs to DASD, ensure that you preallocate the backup version data sets and make them accessible to any processor that you want to be able to back up the CDSs. Because CDS backup is done by the primary processor as the first phase of automatic backup, you must share the preallocated backup version data sets with any secondary processor that you want to back up the CDSs.

Tape CDS backup versions

If you are backing up your CDSs to tape, and your CDSs are not in a shared user catalog, the backup versions will not "roll off" if they are created on one processor and deleted on another processor.

Journal considerations in a multiple DFSMShsm host environment

In a multiple DFSMShsm host environment:

- The journal data set must reside on shared DASD. Before an I/O operation begins, DFSMShsm reserves the volume containing the journal data set. After the I/O operation has completed, DFSMShsm releases the volume.
- If the multiple DFSMShsm hosts share a single set of control data sets, they must also share a single journal. All DFSMShsm recovery procedures are based on a single journal to merge with a backed up version of a control data set.

Monitoring the control and journal data sets in a multiple DFSMShsm host environment

To maintain space-use information for the MCDS, BCDS, and OCDS in a multiple DFSMShsm host environment, all DFSMShsm hosts can access the multiple processor control record maintained in the MCDS. This record accumulates space-use information about the combined activity of all hosts.

Problem determination aid log data sets in a multiple DFSMShsm host environment

In a multiple DFSMShsm host environment, you must allocate unique PDOX and PDOY data set names for each host if the PDA log data sets are cataloged in a shared catalog. Placing the PDA log data sets on a volume managed by another processor can lead to degraded performance or a lockout. Because the PDA log data sets cannot be shared among hosts, you should:

- Catalog the PDA log data sets.
- Allocate two PDA log data sets on the same volume with data set names of *qualifier*.Hx.HSMPDOX and *qualifier*.Hx.HSMPDOY. Substitute the DFSMShsm authorized-user identification for *qualifier* and substitute a unique DFSMShsm host identifier for *x*. These substitutions make the PDA log data set names unique and identifiable with the host. **Example:** If the host identification is A, the PDA log data set names could be:
  - *qualifier*.HA.HSMPDOX
  - *qualifier*.HA.HSMPDOY

DFSMShsm log considerations in a multiple DFSMShsm host environment

The DFSMShsm Log contains a duplicate of information that is kept in DFSMShsm activity logs, SMF data, and DFSMShsm Problem Determination Aid (PDA) trace data. It is optional, and many customers chose not to expend the amount of overhead needed to maintain these files. However, if you decide to have the DFSMShsm log, and you are running in a multiple DFSMShsm host environment, you must allocate unique LOGX and LOGY data set names for each host. Placing the log on a volume managed by another host may lead to degraded performance or a lockout. Because the log data sets cannot be shared among hosts, you should:

- Catalog the log data sets.

- Allocate two log data sets on the same volume with data set names of *qualifier*.HA.HSMLOGX and *qualifier*.HA.HSMLOGY. Substitute the DFSMShsm authorized-user identification for *qualifier* and substitute a unique DFSMShsm host identifier for *n*. These substitutions make the log data set names unique and identifiable with the host. **Example:** If the host identification is 1, the log data set names could be:
  – *qualifier*.HA.HSMLOGX1
  – *qualifier*.HA.HSMLOGY1

Edit log data set considerations in a multiple DFSMShsm host environment

In a multiple DFSMShsm host environment, you must allocate a unique edit log data set for each host if the edit logs are cataloged in a shared catalog. Placing the edit log on a volume managed by another host can lead to degraded performance or lockout. Because the edit log data set cannot be shared among hosts, you should:

- Catalog the edit log data set.
- Allocate the edit log data set with a data set name of *qualifier*.EDITLOG*n*. Substitute the DFSMShsm authorized-user identification for *qualifier* and substitute a unique DFSMShsm host identifier for *n*. These substitutions make each data set name unique and identifiable with the host. **Example:** If the host identification is 1, the edit log data set name could be:
  – *qualifier*.EDITLOG1

Small-data-set-packing data set considerations in a multiple DFSMShsm host environment

Place small-data-set-packing (SDSP) data sets in a different catalog and allocate them on a different volume from the CDSs. Following this guideline prevents an enqueue lockout from occurring when CDS backup starts on one system and migration to an SDSP is already in progress on another system.

SDSP data set share options

You should define a (2 *x*) share option for your SDSP data sets. A (2 *x*) share option allows other programs to process the records in the SDSP data set.

A (3 *x*) share option also allows other programs to update the records of the SDSP data set. However, a (2 *x*) share option is a safer choice, because a (2 *x*) share option allows only DFSMShsm to access the SDSP data sets during DFSMShsm processing. Other jobs cannot update or allocate the SDSP data sets while DFSMShsm is processing. The (2 *x*) option causes VSAM to do an ENQ for the SDSP data set, which provides write-integrity in a single DFSMShsm host environment even though there are non-DFSMShsm jobs processing the SDSP data set. When global resource serialization is in use, the (2 *x*) option default provides write-integrity in a multiple DFSMShsm host environment.

If the (3 *x*) option is used instead, VSAM does not do any ENQs for the SDSP data set.

Maintaining data set integrity

When you implement DFSMShsm in a JES2 or JES3 environment with multiple DFSMShsm hosts, maintain the integrity of your DFSMShsm data sets by:

- Defining only a primary space allocation for the MCDS, BCDS, OCDS, and journal. The journal must be allocated with contiguous space.
- Ensuring that the control data sets, journal data set, and SDSP data sets are cataloged in shared catalogs.
- Ensuring that all volumes managed by DFSMShsm can be shared by all processors and that all data sets on the volumes are cataloged in catalogs that can be shared by all processors.

Serialization of resources

Certain CDS records are used to serialize resources needed by tasks running on different processors. The records used for the serialization have a *host ID* field. When this field in a record contains '00', it is an indication that no processor is currently serialized on the resource.

If DFSMShsm is unable to update a record to remove the serialization, the related resource remains unavailable to DFSMShsm tasks running in other processors until the host ID field of the record is reset. Unavailability of DFSMShsm resources can occur if DFSMShsm or MVS abnormally ends while a record is serialized with its identification. Use the LIST HOST command to list all CDS records containing a specified processor identification in the host ID field. After examining the list, use the LIST command again with the HOST and RESET parameters to reset the host ID field of the record.

Volume considerations in a multiple DFSMShsm host environment

Consider the following when you implement DFSMShsm in a multiple DFSMShsm host environment:

- DFSMShsm does not reserve the volume containing a user's data set if the user issues a request to migrate or back up the data set, but depends upon global serialization of the SYSDSN resource.
- While DFSMShsm calculates the free space of a volume, it reserves the volume. This can interfere, momentarily, with the response time for the functions that require access to that volume.
- Run automatic primary space management, backup, and dump during periods of low system use and low interactive user activity to reduce contention for data sets among processors. When a DFSMShsm-managed volume is being processed by space management or backup in a DFHSMDATASETSERIALIZATION environment, other processors can have performance problems if they attempt to access the volume. To eliminate these performance problems, consider using USERDATASETSERIALIZATION instead, which will require Global Resource Serialization, or similar product, if data is shared among multiple z/OS images.
- You can run automatic secondary space management (SSM) in multiple tasks. Doing so can avoid contention for SDSP data sets if you run secondary space management during primary space management. For a discussion of SDSP data set contention, see "Multitasking considerations for SDSP data sets" on page 53.

JES3 considerations

In a JES3 environment, the SMS volumes and the non-SMS, DFSMShsm-managed volumes in the DFSMShsm general pool must be shared by all processors. Furthermore, if some of the volumes in a data set or volume pool are also in the general pool, all volumes in both pools must be shared.

Running automatic processes concurrently in a multiple DFSMShsm host environment

The first phase of automatic backup can include the backup of DFSMShsm control data sets on the primary processor. Exclusive serialization ensures that the CDSs are not changed while DFSMShsm is backing them up. After they have been backed up, exclusive serialization is changed to shared serialization. Because the automatic primary space management, automatic secondary space management, and automatic dump processes can change records, they do shared serialization on the control data sets.

During automatic volume processing, DFSMShsm skips over a DFSMShsm-managed volume currently being processed anywhere in the configuration sharing the DFSMShsm control data sets. DFSMShsm then retries those skipped DFSMShsm-managed volumes after processing the rest of the volumes. DFSMShsm tries processing the skipped volumes up to nine times. DFSMShsm waits for five minutes between volume process attempts if it does not process any skipped volumes during the retry loop through the list. If a volume is not processed, an error message is written to the function's activity log.

Commands that cause changes to control data sets do not run while the control data sets are being backed up and are suspended until the control data set backup completes.

Multitasking considerations in a multiple DFSMShsm host environment

When you run multiple tasks in a multiple DFSMShsm host configuration, consider the most effective use of your z/OS image and the number of DFSMShsm hosts running within each of those z/OS images. For example, you might ask whether it is more efficient to perform eight tasks with one z/OS image or four tasks with two z/OS images. The answer is that, for space management and backup, it is generally better to perform eight tasks with one z/OS image and distribute those tasks across multiple hosts running in that z/OS image. The reason that one z/OS image offers better performance than two is a result of the DFSMShsm CDS sharing protocol.

For performance reasons, run several migration or backup tasks in one z/OS image versus running a few tasks in each of multiple host environment; however for CDSSHR=RLS, it is better to spread out the tasking across multiple DFSMShsms.

Performance considerations in a multiple DFSMShsm host environment

Review the following performance consideration if you have configured your system to use a multiple DFSMShsm host environment:

- ARCGPA and ARCCAT resources are held when a DFSMShsm function requires an update to or read of a control data set (CDS) record. While the resources are held, all other functions must wait for the controlling function to complete its task. In some cases, the wait is long and can delay new DFSMShsm request or functions from starting. For example, when a CDS backup task appears to be hung, it might be waiting for a large data set recall task to complete. Furthermore, new recall requests wait for both the original recall request and the CDS backup to complete before being processed. Specifically:

- In a record level sharing (RLS) environment, a CDS backup must wait for all DFSMShsm functions across the HSMplex to complete.
- In a non-RLS environment, a CDS backup must wait for all DFSMShsm functions within the same LPAR as the DFSMShsm host performing the CDS backup to complete.

To minimize this delay, DFSMShsm hosts in the same HSMplex (RLS environment) or within the same LPAR (non-RLS environment) allow CDS backup to begin immediately after all pending CDS updates are complete. The host performing the CDS backup sends a notification to the other hosts using cross-system coupling facility (XCF) services. All hosts processing functions and tasks that might delay CDS backup complete pending CDS updates, suspend new CDS updates, and release all resources (such as ARCGPA and ARCCAT) necessary for CDS backup to begin. After CDS backup is complete, suspended functions and tasks resume.

- Because all DFSMShsm activity is quiesced during quiesced journal backup, user and production jobs that require recall or recovery resources wait until the entire CDS and journal backup is complete. Therefore, the impact of journal backup on DFSMShsm availability and performance should be considered when planning a backup schedule.

  Non-intrusive journal backup can be used to reduce the impact on DFSMShsm availability and performance when concurrent copy is available for CDS backup and the SETSYS JOURNAL(RECOVERY) setting is in effect. This method of journal backup does not hold resources during journal backup, which allows DFSMShsm activity to continue. Resources are held only at the end of journal backup while changes to the journal that occurred during creation of the initial backup copy are appended to the backup copy. The control data sets are then backed up using concurrent copy. For more information about the non-intrusive journal backup method, see the topic about using non-intrusive journal backup in *z/OS DFSMShsm Storage Administration*.
- When there are multiple HSMplexes within a z/OS sysplex, the startup procedure keyword PLEXNAME should be specified. For more information about the SETSYS PLEXNAME command, see the SETSYS command in *z/OS DFSMShsm Storage Administration*.

MASH configuration considerations

When running in a multiple address space DFSMShsm (MASH) configuration where XCF services are not available, the following should be considered:

- Long running functions on other DFSMShsm hosts (including those that start before automatic CDS backup) will not release the resources required to allow CDS backup to obtain an exclusive lock. This prevents starting CDS backup.
- The enqueue prevents all data set or volume-type functions from accessing the control data sets while they are being backed up. Therefore, you should carefully consider the best time to start backup of the control data sets.

Part 2. Customizing DFSMShsm

The following information is provided in this topic:

- **Chapter 12, "DFSMShsm in a sysplex environment," on page 279** describes information about using DFSMShsm in a sysplex environment. It covers information about how to promote secondary hosts, how to use extended addressability for control data sets, and how DFSMShsm functions in a GRSplex.
- **Chapter 13, "Calculating DFSMShsm storage requirements," on page 293** provides information about customizing your computing system storage for DFSMShsm, including storage calculation work sheets.
- **Chapter 14, "DFSMShsm libraries and procedures," on page 297** describes how and where to create DFSMShsm procedures and parameter library members.
- **Chapter 15, "User application interfaces," on page 313** describes information about DFSMShsm application programs and how to invoke them.
- **Chapter 16, "Tuning DFSMShsm," on page 329** describes information that you can use to tune DFSMShsm through use of DFSMShsm-supported patches.
- **Chapter 17, "Special considerations," on page 373** describes information that you should consider before you install DFSMShsm.
- **Chapter 18, "Health Checker for DFSMShsm," on page 383** describes information about Health Checker for DFSMShsm.

Chapter 12. DFSMShsm in a sysplex environment

To facilitate managing several z/OS images at once, the z/OS SYStems comPLEX, or *sysplex*, allows simplified multisystem communication between systems without interference. A sysplex is a collection of z/OS images that cooperate, using certain hardware and software products, to process workloads. The products that make up a sysplex provide greater availability, easier systems management, and improved growth potential over a conventional computer system of comparable processing power.

Types of sysplex

There are two types of sysplex: base and parallel.

- A *base sysplex* is a sysplex implementation *without* a coupling facility.
- A *parallel sysplex* is a sysplex implementation *with* a coupling facility.

Systems in a *base* sysplex communicate using channel-to-channel (CTC) communications. In addition to CTC communications, systems in a *parallel* sysplex use a coupling facility (CF), which is a microprocessor unit that enables high performance sysplex data sharing. Because parallel systems allow faster data sharing, workloads can be processed more efficiently.

For more information about sysplexes, refer to Parallel Sysplex Overview (http://www.ibm.com/systems/z/advantages/pso/sysover.html).

Sysplex support

If you are running DFSMShsm in a sysplex environment, the following functions can greatly enhance your ability to successfully manage that environment:

Single GRSplex Serialization
: Allows each HSMplex, within a single GRSplex, to operate without interfering with any other HSMplex. See "Single GRSplex serialization in a sysplex environment" on page 280.

Secondary Host Promotion
: Allows one DFSMShsm host to automatically assume the unique functions of another DFSMShsm host that has failed. See "Secondary host promotion" on page 283.

Control Data Set Extended Addressability
: Allows CDSs to grow beyond the 4 GB size limit. See "Control data set extended addressability in a sysplex environment" on page 289.

Record Level Sharing
: Allows CDSs to be accessed in record level sharing (RLS) mode. See "Using VSAM record level sharing" on page 32.

Common Recall Queue
: Balances the recall workload among all hosts in an HSMplex by implementing a common queue. See "Common recall queue configurations" on page 289.

CDS Backup Contention Notification
: Allows a CDS Backup host to communicate via XCF to other hosts in an HSMplex that they should release the necessary resources to allow CDS Backup to begin.

Single GRSplex serialization in a sysplex environment

One or more processors with DFSMShsm installed and running that share a common MCDS, OCDS, BCDS, and journal is called an *HSMplex*. One or more MVS systems that use global serialization to serialize access to shared resources (for example, data sets on shared DASD volumes) is called a *GRSplex.*

If two HSMplexes exist within a sysplex environment, one HSMplex interferes with the other HSMplex whenever DFSMShsm tries to update CDSs in non-RLS mode or when it is performing other functions, such as level 1 to level 2 migration. This interference occurs because each HSMplex, although having unique resources, uses the same resource names for global serialization.

Within a GRSplex, you can now place multiple HSMplexes into a single GRSplex. The single GRSplex serialization function allows DFSMShsm to translate minor global resource names to unique values within the HSMplex, thus avoiding interference between HSMplexes.

Resource serialization in an HSMplex environment

All DFSMShsm hosts within an HSMplex must use the same translation technique. If a host detects an inconsistency in the translation technique, the detecting host immediately shuts down.

The new startup keyword RNAMEDSN specifies whether you want to keep the old translation technique or use the newer technique. The RNAMEDSN keyword directs DFSMShsm to perform the new translation technique.

For CDS resource serialization considerations, see “Preventing interlock of DFSMShsm control data sets” on page 270.

Enabling single GRSplex serialization

You can specify whether you want to upgrade your system to the new translation technique by using the keyword RNAMEDSN in the startup procedure.

```
RNAMEDSN = YES | NO
```

When you specify YES, DFSMShsm invokes the new method of translation, which uses the CDS and journal data set names. When you specify NO, you are saying that you want to continue to keep the old method of translation; however, in a sysplex with multiple HSMplexes, one HSMplex may interfere with another. The default for the RNAMEDSN keyword is NO.

Identifying static resources

Table 44 on page 281 shows those resources (when RNAMEDSN=NO) that, when obtained, cause one HSMplex to interfere with another.

Table 44. Global Resources, Qname=ARCENQG

Major (qname) Resource Name	Minor (rname) Resource Name	Serialization result
ARCENQG	ARCBMBC	Enqueues during the attach of ARCBMBC subtask, which moves backup copies from ML1 to backup tapes.
	ARCCDSVF	Serializes a CDS backup function to ensure that only one CDS backup is running within one HSMplex.
	ARCCDSVD	Enqueues while copying CDSVDATA.
	ARCL1L2	Enqueues L1 to L2 migration. L1 to L2 migration is a function of secondary space management.
	ARCMCLN	Enqueues migration cleanup. This is part of secondary space management.
	RECYC_L2	Prevents two hosts from recycling ML2 tapes concurrently.
	RECYC_SP	Prevents two hosts from recycling backup spill tapes concurrently.
	RECYC_DA	Prevents two hosts from recycling backup daily tapes concurrently.
	ARCBCDS ARCMCDS ARCOCDS	Enqueues CDSs (not obtained in RLS mode). Note: IF CDSQ is specified, then ARCGPA, ARCxCDS, SYSTEMS, SHARE translates to ARCENQG, ARCxCDS, SYSTEMS, EXCLUSIVE.
	ARCCAT	In RLS mode, enqueues change from ARCGPA/ARCCAT STEP to ARCENQG/ARCCAT SYSTEMS to prevent CDS updates during CDS backup.
	HOST||Hostid	Ensures that only one host is started with this host identifier.
	EXPIREBV	Ensures that only one EXPIREBV command is running within HSMplex.
	COPYPOOL||*cpname*	SYSTEMS enqueue **Note:** The scope of the fast replication copy pool extends beyond an HSMplex because a copy pool is defined at the SMSplex level. All DFSMShsm hosts, regardless of which HSMplex they reside in, are prevented from processing the same copy pool. The resource is obtained unconditionally and if the resource is not immediately available, it waits.
	CPDUMP||*cpname*||V*nnn*	SYSTEMS enqueue **Note:** The scope of the fast replication copy pool extends beyond an HSMplex because a copy pool is defined at the SMSplex level. All DFSMShsm hosts, regardless of which HSMplex they reside in, are prevented from processing the same copy pool. The resource is obtained unconditionally and if the resource is not immediately available, it waits.
ARCGPA	ARCRJRN	This is the volume reserve of the journal volume.
ARCBTAPE	volser.TAKEAWAY	Allows Recover Tape Takeaway.
ARCBTAPE	volser	Allows Recover Tape Takeaway.

Translating static resources into dynamic resources

If you have enabled DFSMShsm to use the translation technique specified by RNAMEDSN=YES, the minor name (or resource name) will be translated to a new minor name:

function&cdsdatasetname

where

- *function* is the current Rname (such as ARCL1L2)
- *cdsdatasetname* is the base cluster name of the control data set associated with the function that is being serialized (such as, MCDS for L1 to L2 migration), or the CDS itself that is being serialized

Rule: The ampersand (&) between `function` and `cdsdatasetname` is a required character. You must type the ampersand as shown.

Table 45 lists all of the translated resource names (when RNAMEDSN=YES).

Table 45. Rname Translations

Current Rname	Translated Rname
ARCBMBC	ARCBMBC&bcdsdsn
ARCCDSVF	ARCCDSVF&mcdsdsn
ARCCDSVD	ARCCDSVD&mcdsdsn
ARCL1L2	ARCL1L2&mcdsdsn
ARCMCLN	ARCMCLN&mcdsdsn
RECYC_L2	RECYC_L2&ocdsdsn
RECYC_SP	RECYC_SP&ocdsdsn
RECYC_DA	RECYC_DA&ocdsdsn
ARC*x*CDS	ARC*x*CDS&cdsdsn
ARCCAT	ARCCAT&mcdsdsn
ARCRJRN	ARCRJRN&jrnldsn
HOST||Hostid	HOST||Hostid&mcdsdsn
EXPIREBV	EXPIREBV&bcdsdsn
volser	volser&bcdsdsn
volser.TAKEAWAY	volser.TAKEAWAY&bcdsdsn

Compatibility considerations

Consider the following coexistence issues before you run DFSMShsm within an HSMplex:

- If all DFSMShsm hosts within one HSMplex are running at DFSMS/MVS Version 1 Release 5.

  All DFSMShsm hosts must use the same serialization method. If not, at least one of the hosts will shut down (that is, each host detecting a mismatch will shut down).
- Not all DFSMShsm hosts within one HSMplex are running at DFSMS/MVS Version 1 Release 5.

  If an HSMplex has both Version 1 Release 5 and pre-Version 1 Release 5 running concurrently, then the Version 1 Release 5 hosts cannot specify RNAMEDSN=YES. If RNAMEDSN=YES is specified, hosts that detect the mismatched serialization method will shut down.
- If two or more HSMplexes are running concurrently.

  Each HSMplex using an old serialization method will interfere with other HSMplexes. HSMplexes using the new serialization method will not interfere with other HSMplexes. However, in a two-HSMplex environment, one can use the old method and the other can use the new method; neither one will interfere with the other.

Secondary host promotion

DFSMShsm allows secondary hosts to take over functions for a failed primary host. This failure can be either an address space failure or an entire z/OS image failure. In addition, DFSMShsm allows another host to take over secondary space management (SSM) from a failed host, which can either be the primary or a secondary host. Secondary host promotion ensures continuous availability of DFSMShsm functions. Host promotion occurs without users who have to interact with other programs, receive or interpret console messages, or issue commands from batch jobs.

The following definitions are key to understanding the concept of secondary host promotion:

- An **original host** is a host that is assigned to perform primary host or SSM responsibilities.
- A **secondary host** is a host that is not assigned to perform primary host or SSM responsibilities.
- A **primary host** is a host that performs primary level functions.

  The **primary host** is the only host that performs the following functions:
  - Hourly space checks (for interval migration and recall of non-SMS data)
  - During autobackup: Automatic CDS backup
  - During autobackup: Automatic movement of backup versions from ML1 to tape
  - During autobackup: Automatic backup of migrated data sets on ML1
  - During autodump: Expiration of dump copies
  - During autodump: Deletion of excess dump VTOC copy data sets
- An **SSM host** is generally the only host that performs SSM functions.
- A host is said to be **promoted** when that host takes over the primary or SSM (or both) host responsibilities from an original host.
- A host is said to be **demoted** when it has had its primary or SSM (or both) host responsibilities taken over by another host. There is always a corresponding promoted host for each demoted host, and vice versa.

Enabling secondary host promotion from the SETSYS command

For either a base or parallel sysplex, DFSMShsm, using XCF, can enable secondary hosts to take over any unique functions that are performed by the failed primary host. There can be three types of failures:

- DFSMShsm placed in emergency mode
- DFSMShsm address space failures
- Entire z/OS image failures

Likewise, another host within an HSMplex can assume the responsibilities of any host (either the primary or secondary host) that is performing secondary space management, if the host performing SSM fails.

Rule: To enable secondary host promotion, you must configure XCF on the active DFSMShsm system. DFSMShsm must be running in multisystem mode.

To enable secondary host promotion, specify the SETSYS PROMOTE command with either or both of the following parameters:
- PRIMARYHOST(YES|NO)
- SSM(YES|NO)

where

PRIMARYHOST(YES)
: You want this host to take over primary host responsibilities for a failed host.

PRIMARYHOST(NO)
: You do not want this host to take over primary host responsibilities for a failed host.

SSM(YES)
: You want this host to take over the SSM responsibilities for a failed host.

SSM(NO)
: You do not want this host to take over the SSM responsibilities for a failed host.

Note:
1. NO is the default for both SETSYS PROMOTE parameters (PRIMARYHOST and SSM).
2. Only those DFSMShsm hosts running on DFSMS/MVS Version 1 Release 5 and above are eligible to use secondary host promotion functions.
3. This parameter is ignored when the system is running in LOCAL mode. If the system is running in MONOPLEX mode, the secondary host promotion function is active, but is unable to perform actions because cross-host connections are not enabled.
4. An SSM host is not eligible to be promoted for another SSM host.
5. PRIMARYHOST(YES) is ignored if it is issued on the primary host.
6. The SETSYS command does not trigger promotion. That is, a host can only be eligible to be promoted for hosts that fail after the SETSYS command has been issued.
7. Do not make a host eligible for promotion if its workload conflicts with responsibilities of the original host or if it is active on a significantly slower processor.

Configuring multiple HSMplexes in a sysplex

If you have multiple HSMplexes in a sysplex, you must use the SETSYS keyword PLEXNAME in a ARCCMD*xx* member of SYS1.PARMLIB.

SETSYS PLEXNAME(*HSMplex_name_suffix)*

The PLEXNAME keyword distinguishes the separate HSMplexes within a single sysplex. If you have only one HSMplex in a sysplex, you can use the default name. The default name is ARCPLEX0: the suffix is PLEX0, with a prefix of ARC.

If you specify an HSMplex name other than the default on one host, you must also specify that name on all other DFSMShsm hosts in that HSMplex.

Additional configuration requirements for using secondary host promotion

The following requirements apply to the use of secondary host promotion:

- If the ARCCBEXT exit is used by the primary host, it must be available for use on all hosts eligible to be promoted for the primary host. If the ARCMMEXT exit is used by the SSM host, it must be available for use on all hosts eligible to be promoted for the SSM host.
- The CDS backup data sets must be cataloged on all systems that are eligible to be promoted for primary host responsibilities.
- In a multisystem environment, DFSMShsm always sets the option to NOSWAP.

When a host is eligible for demotion

Any one of the following conditions will initiate the demotion process:

- The primary or SSM host goes into emergency mode
- The primary or SSM host is stopped with the DUMP or PROMOTE keyword
- The primary or SSM host is stopped while in emergency mode
- The primary or SSM host is canceled, DFSMShsm fails, or the system fails
- A promoted host is stopped or fails by any means

Note:

1. If an active primary or SSM host has been demoted and it has not taken back its responsibilities (for example, it is in emergency mode), then you can invoke any type of shutdown and the host will remain in its demoted status.
2. Do not change the host ID of a host that has been demoted.

How secondary host promotion works

When a primary or SSM host becomes disabled, all DFSMShsm hosts in the HSMplex are notified through XCF. Any host that is eligible to perform the functions of the failed host will attempt to take over for the failed host. The first host that successfully takes over for the failed host becomes the promoted host. There is no means available for assigning an order to which hosts take over the functions of a failed host.

Note: Secondary host promotion is designed to occur when the primary host fails or becomes unexpectedly disabled. To cause secondary host promotion during a normal shutdown of DFSMShsm, issue the STOP command with the PROMOTE or DUMP parameters.

If an original host is both a primary and an SSM host, its responsibilities can be taken over by two separate hosts.

Example: If a secondary host specifies

SETSYS PROMOTE(PRIMARYHOST(YES) SSM(NO))

and a different secondary host specifies

SETSYS PROMOTE(PRIMARYHOST(NO) SSM(YES)),

then it is possible for each host to take over part of the original failed host's work.

Likewise, if a secondary host is eligible to be promoted for both primary and SSM host responsibilities, then it can be promoted for two separate hosts.

Example: Host A, the primary host, and host B, an SSM host.

If the promoted host itself fails, then any remaining host that is eligible for promotion will take over. If additional failures occur, promotion continues until there are no remaining hosts that are eligible for promotion.

If a secondary host fails while it is promoted for an original host and there are no remaining active hosts eligible for promotion, then of any of the secondary hosts that become reenabled before the original host does, only that host that was last promoted for the original host can become the promoted host.

Rule: For secondary host promotion to work at its highest potential, do not use system affinity. All systems must have connectivity to all storage groups. If system affinity is used, then storage groups that are only associated with one system would not be processed when that system was not available.

Promotion of primary host responsibilities

When a secondary host is promoted for a primary host, the secondary host takes over the six unique primary host functions. To do this, the secondary host indicates that it is now the primary host. It copies the automatic backup window and cycle, the status of the ARCCBEXT exit, and the auto backup restart variables from the primary host. Messages ARC0154I and ARC0271I are issued to notify the user of the updates to the window and cycle. The secondary host also copies the automatic dump window and cycle from the primary host and then issues messages ARC0638I and ARC0273I. Message ARC1522I is issued to notify the user of the promotion.

How auto functions affect secondary host promotion

The following scenarios can occur for automatic functions (backup and dump) as the result of a demotion:

Promoted host is not an automatic backup host: If the promoted host is *not* an automatic backup host, it will only perform the unique primary host automatic backup functions during the automatic backup window. It will not backup managed volumes. If the promotion occurred while the original primary host was performing one of the three unique autobackup functions, the promoted host takes over from the point where the original host left off.

Example: If the original primary host had just completed backing up the CDSs before it failed, then the promoted host will not backup the CDSs again, but will begin by moving data set backup versions from ML1 to tape.

Promoted host is an automatic backup host: If the promoted host *is* an automatic backup host, it performs the three unique primary host automatic backup functions before it performs backups of managed volumes. If the promotion occurred while the original primary host was performing one of the three unique autobackup functions, then whether this host takes over from where the original primary host left off depends on its own automatic backup window. If this host's window overlaps the original primary host's window and this host has also begun performing automatic backup, then it will *not* pick up from the point where the original primary host left off, but it will continue performing backups of managed volumes. The unique level functions that were not completed by the original primary host will not be completed until the next automatic backup window. If this host’s automatic backup window is such that it was not performing automatic backup when it was promoted, then it will take over from where the original primary host left off.

Configuring automatic backup hosts in an HSMplex: If you are using secondary host promotion, take special care when you are configuring automatic backup in an HSMplex.

- First, there should be more than one automatic backup host. This ensures that volume backups of managed volumes are performed even when the primary host is disabled.

  Note: Promoted hosts only take over *unique* functions of the original host. They do not take over functions that can be performed by other hosts.
- Second, if a secondary automatic backup host is eligible to be promoted for the primary host, then its backup window should be offset from the original primary host’s window in a way that it can take over from where the original primary host left off.

  Example: Its start time could correspond with the average time that the primary host finishes its unique automatic backup functions.

Note: These scenarios assume that the primary host is always an automatic backup host.

Promoted host is not an automatic dump host: If the promoted host is not an automatic dump host, it can only perform the two unique primary host automatic dump functions during the automatic dump window. It does not perform volume dumps. If promotion occurs while the original primary host was performing automatic dump functions, this host does *not* restart from where the original primary host left off, but starts automatic dump from the beginning.

Promoted host is an automatic dump host: If the promoted host is an automatic dump host, it will perform the two unique automatic dump functions in addition to performing volume dumps. If promotion occurs while the original primary host was performing automatic dump functions, this host will restart autodump from the beginning, if it was *not* already performing autodump in its own window. If it was already performing autodump in its own window, then it will only perform the unique functions if it has not already passed their phases in the window. Otherwise, it will not perform the unique functions until the next window.

Configuring automatic dump hosts in an HSMplex: It is recommended that there be more than one automatic dump host in an HSMplex. This ensures that volume dumps are performed even if the primary host is disabled.

Promotion of SSM host responsibilities

When a non-SSM host is promoted to take over the functions for an SSM host, the SSM window and cycle, the status of the ARCMMEXT exit, and SSM restart variables from the original SSM host are copied to the promoted host. Messages ARC0151I and ARC0273I notify the user of the window or cycle updates. Message ARC1522I notifies the user of the promotion, and the promoted host performs all secondary space management functions. If promotion occurs during the SSM window, then the promoted host attempts to restart SSM functions as close to where the original SSM host left off as possible.

If there is more than one SSM host, all of them are eligible to have their responsibilities taken over by other hosts; however, SSM hosts are not eligible to be promoted for other SSM hosts. The number of hosts that can be demoted at any one time is limited by the number of hosts that are eligible to be promoted.

How the take back function works

When an original host is re-enabled to perform its unique responsibilities (through a restart or by leaving emergency mode), the take back process begins. The take back process involves the following procedures:

- The promoted host recognizes that the original host is enabled and gives up the responsibilities that it took over.
- Until the promoted hosts give them back, the original host does not perform any of the responsibilities that were taken over by the promoted hosts.

The following scenarios pertain to the take back function:

The promoted host gives up the promoted responsibilities: When a promoted host recognizes that the original host is once again eligible to perform its unique responsibilities, it gives up the functions that it first took over. It resets its windows, its cycles, and its exit settings to the values that existed before it became promoted.

Attention: Any changes that were made to the window and the cycle while the host was promoted are lost.

If the original host becomes enabled while the promoted host is performing one of the functions that it was promoted for, then the promoted host continues performing that function to its completion. After a promoted host has given up the promoted functions, it is immediately available for promotion again.

The original host waits for the promoted host: An original host cannot take back its unique responsibilities that have been taken over until the promoted host gives them up. If the original host becomes enabled during a window of a function that the promoted host is currently performing, then the original host does not perform the unique functions that have been taken over. Once the promoted host has given up the unique responsibilities, the original host takes those responsibilities back and resumes normal processing.

Example: If it restarts during the autobackup window, the original primary host will not perform the three unique autobackup functions, but will start with volume backups. If the promoted host has already completed the function for the current window, then the original host will not perform the function until the next window.

Emergency mode considerations

If you want to restart DFSMShsm in emergency mode, consider the following conditions:

- To restart a **demoted** host in emergency mode, specify the EMERGENCY parameter in your startup procedures to avoid the window between the time a demoted DFSMShsm host joins XCF and attempts to take back its functions and the time—after setup—that the SETSYS EMERGENCY command is issued.
- If an **original** host restarts in emergency mode, it will not take back its level functions.
- A host in emergency mode cannot promote itself.

Considerations for implementing XCF for secondary host promotion

The cross system coupling facility (XCF) component of MVS/ESA provides simplified multisystem management. XCF services allow authorized programs on one system to communicate with programs on the same system or on different systems. If a system ever fails, XCF services allow the restart of applications on this system or on any other eligible system in the sysplex.

Before you configure the cross-coupling facility (XCF) in support of the secondary host promotion function, consider the following information:

- There will be only one DFSMShsm XCF group per HSMplex. The XCF group name is the HSMplex name, with the default name being ARCPLEX0.
- There will be one XCF group member for each DFSMShsm host in the HSMplex.
- DFSMShsm does not use the XCF messaging facilities.

For more information about configuring XCF in a sysplex, refer to the following publications:

- *z/OS MVS Setting Up a Sysplex*
- *z/OS MVS Programming: Sysplex Services Guide*
- *z/OS MVS Programming: Sysplex Services Reference*
- *z/OS MVS Programming: JES Common Coupling Services*
- *z/OS MVS System Commands*

Control data set extended addressability in a sysplex environment

As it becomes possible to combine more HSMplexes into a single HSMplex, it also becomes more likely that CDS sizes will grow beyond the 16 GB size for MCDS and BCDS data sets and 4 GB size for OCDS data sets. VSAM extended addressability is a function that allows you to define each CDS, so that the CDSs can grow beyond those initial limitations.

Using VSAM extended addressability in a sysplex

DFSMShsm supports VSAM KSDS extended addressability capability that uses the following access modes for its CDSs: record level sharing (RLS)access, CDSQ serialization, or CDSR serialization.

Extended addressability considerations in a sysplex

The following considerations or requirements may affect extended addressability for your CDSs:

- Mixing EF clusters and non-EF clusters is permissible because each cluster is treated as a separate entity. However; if any cluster is accessed in RLS mode, then all clusters must be accessed in RLS mode.
- Because EF data sets may contain compressed data, DFSMShsm issues warning message ARC0130I (RC16)whenever it detects this condition. RC16 means that a given CDS contains compressed data, which may affect performance.

Common recall queue configurations

For an overview of the CRQ environment, refer to the *z/OS DFSMShsm Storage Administration*.

A standard HSMplex configuration is one where all hosts are connected to the same CRQ and all hosts are eligible to process recalls. The use of a CRQ enables the following alternative configurations:

Recall Servers
: Certain hosts may be configured to process all recalls, while other hosts only accept recall requests. Figure 80 on page 291 is a graphic overview of a CRQplex in which several hosts are configured to process recall requests, while one host is configured to only accept recall requests. You can use the DFSMShsm HOLD command to configure a host to not select recall requests.

  Example: On the hosts that you want to only accept recall requests, issue the HOLD COMMONQUEUE(RECALL(SELECTION)) command. These hosts will place recall requests on the CRQ but will not process them.

  When used in conjunction with multiple address space DFSMShsm, this CRQ support can increase the total number of concurrent recall tasks in a z/OS image. Without a CRQ, only the main host can process implicit recalls. When a CRQ environment is established, then all DFSMShsm address spaces in that image can process recall requests. This has the effect of increasing the number of recall tasks from 15 to $n\times15$. (Where n is the number of DFSMShsm address spaces on the z/OS image).

Non-ML2 tape only
: If certain hosts are not connected to tape drives, they can be configured to accept all recall requests but only process those requests that do not require ML2 tape. If you specify the HOLD RECALL(TAPE) command, these hosts only select recall requests from the CRQ that do not require ML2 tape.

Nonparticipating hosts
: If a host within an HSMplex is not participating in CRQ activities, then it places all requests on its local queue and only processes those requests. It performs standard recall take away from recall processing with other hosts that use the CRQ.

If there are hosts within an HSMplex that have data that cannot be shared between systems, then those hosts should not share the same CRQ. If there are sets of hosts within an HSMplex that cannot share data, then each of those sets of hosts can share a unique CRQ so that there are multiple CRQplexes within a single HSMplex. For example, test systems and production systems that are within the same HSMplex, but have data that they cannot share.

Note: While it is possible to maintain multiple disjoint CRQplexes among hosts that share data within a single HSMplex, such a configuration is discouraged. Most of the benefits of this support are achieved as the number of participating hosts increases.

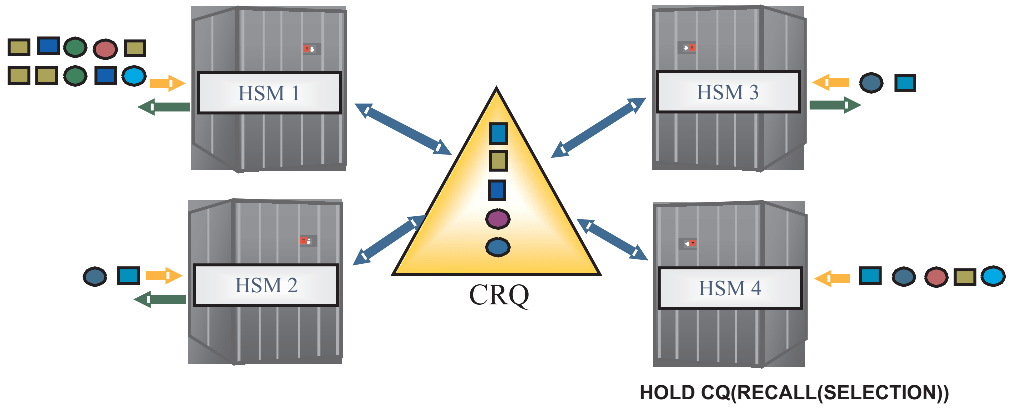


Figure 80. Overview of CRQplex Recall Servers

Common dump queue configurations

In a standard HSMplex configuration, all hosts are connected to the same common dump queue (CDQ) and all hosts are eligible to process dumps regardless of which host was used to submit the requests. The CDQ is a queue of dump requests that is shared by these host, managed by a master scheduler (MS) host and implemented through the use of the cross-system coupling facility (XCF) for host-to-host communication between an XCF defined group and its members. The purpose of the CDQ is to balance dump processing across the resources available in all the hosts and return results back to the host where the request originated to post the user complete.

As illustrated in Figure 81 on page 292, the CDQ group allows for flexible configurations. This provides the capability to:

- Define multiple queues in the same HSMplex
- Allow group members to both receive and process requests, only process requests, or only receive requests.

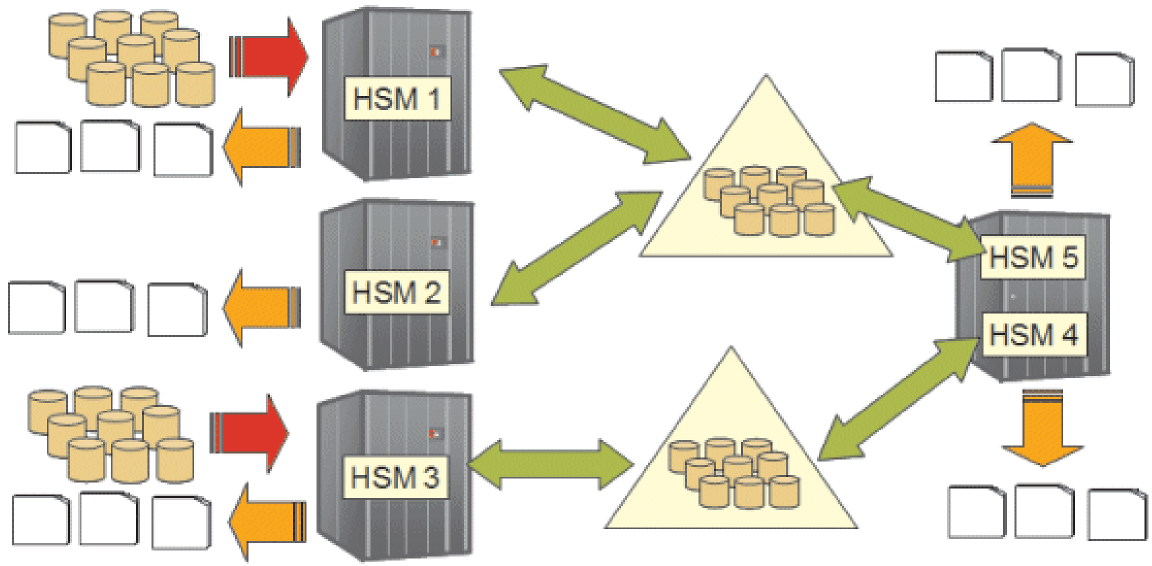


Figure 81. CDQ -- Flexible Configurations

Results are returned to the *submitting host*, but progress and status messages are recorded on the *processing host*, that is, the host processing the request.

The DFSMShsm host types are:

Submitting host
: Receives requests from commands and sends it to the master scheduler host. When the command is completed by the group, the submitting host is notified to post the user that the command has completed.

Master scheduler (MS)
: Is the DFSMShsm host that manages all of the dump requests in the CDQ. It accepts requests from a submitting host and from itself. The master scheduler assigns the requests to eligible hosts (processing hosts), including itself, that have available tasks to process the work, while balancing the utilization of the dump tasks in the group. The master scheduler also manages the interaction between the processing host for stacking and the submitting host for the command complete notifications.

Processing host
: Receives assigned work requests from the master scheduler, completes the work, and interacts with the master scheduler to manage stacking

Any host in the CDQ could be any or all of the host types depending on your environment.

If you do not want dump tasks on a host to be used by the CDQ group, avoid using HOLD DUMP. Instead use SETSYS MAXDUMPTASKS(0) to prevent the host's dump tasks from being used. This has the same effect as HOLD DUMP without the risk of affecting master scheduler responsibilities. HOLD DUMP from the master scheduler prevents it from assigning and processing requests for the CDQ.

Chapter 13. Calculating DFSMShsm storage requirements

The DFSMShsm program requires two categories of storage: common service area (CSA) storage and DFSMShsm address space.

Figure 82 represents an overview of the MVS storage environment.

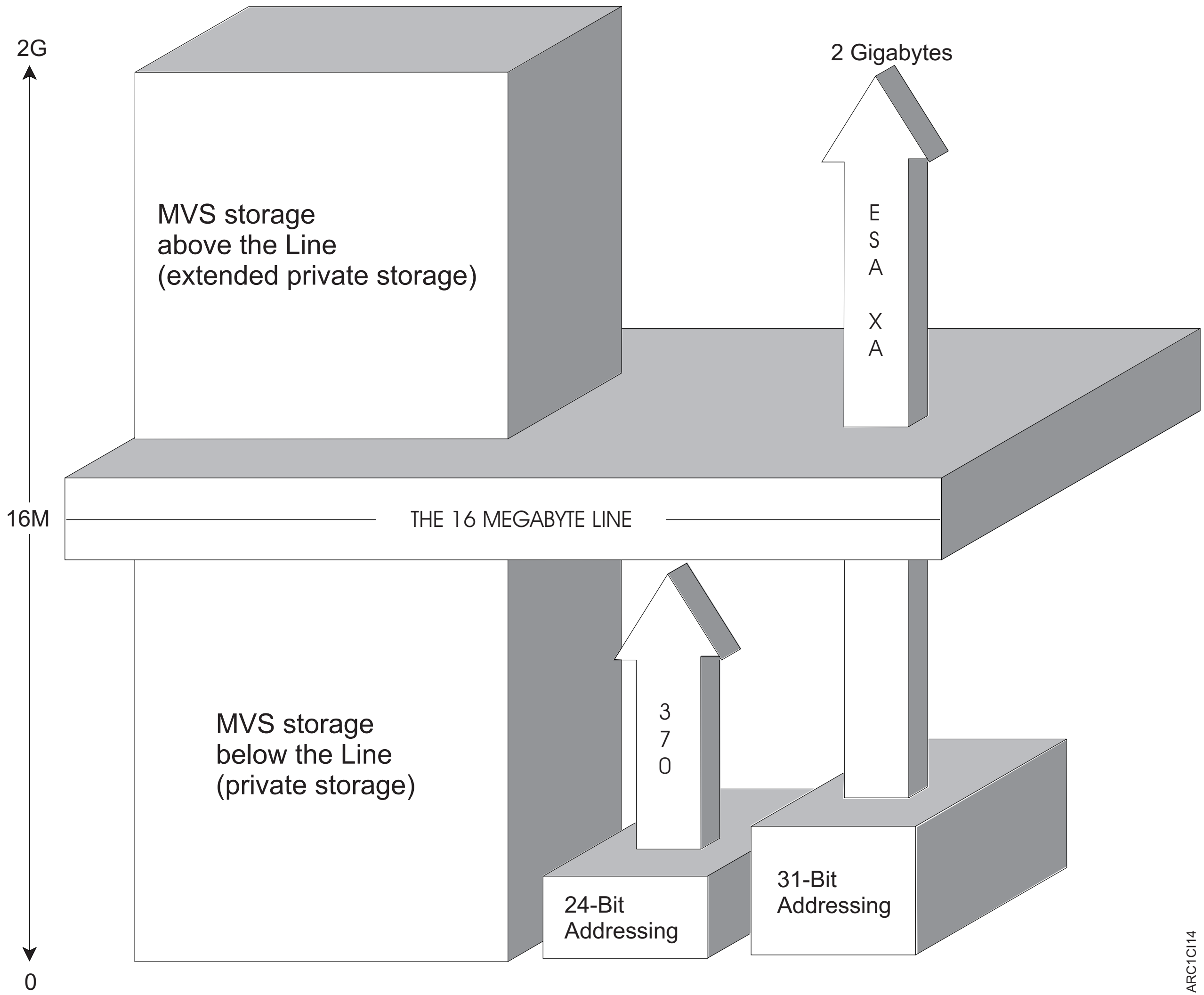


Figure 82. The MVS Storage Environment

DFSMShsm address spaces

The private (24-bit) and extended private (31-bit) address space requirements for DFSMShsm are dynamic. DFSMShsm's region size should normally default to the private virtual address space (REGION=0).

To run ABARS processing, each secondary address space for aggregate backup or aggregate recovery requires 4MB. One megabyte of this ABARS secondary address space is above the line (in 31-bit extended private address space). The other 3MB are below the line (in 24-bit address space).

As you add more functions and options to the DFSMShsm base product, the region-size requirement increases. You should therefore include the maximum region size in your setup procedure.

Storage estimating considerations

If DFSMShsm ends abnormally because it does not have enough virtual storage available (ABEND codes S878, S80A, or S106-C), take corrective action by decreasing the level of multitasking. This is especially true for the number of backup tasks, migration tasks, and dump tasks.

If the maximum region size is not requested, a smaller region size can be used with a corresponding decrease in the level of multitasking. It is best not to specify a region size, allowing DFSMShsm to use all the space that is available.

Storage guidelines

These storage guidelines help you to determine the approximate amount of below-the-line storage that DFSMShsm requires. Table 46 lists the storage requirements for load modules, storage areas, and other DFSMShsm tasks.

Table 46. Below-the-line storage requirements for load modules, storage areas, and DFSMShsm tasks

Load module, storage area, or DFSMShsm task	Storage requirement
DFSMSdss load module	1412KB
System storage	512KB
DFSMShsm static storage	250KB
Each TAPECOPY task	200KB
Each recall task	96KB
Each migration task except secondary space management (SSM) tasks	80KB
SSM control task	62KB + 20(n + 10) bytes, where *n* is the number of ML1 volumes.
Each SSM migration cleanup task	55KB
Each SSM ML1 to ML2 movement task	97KB
Each backup task	80KB
Each autodump task	80KB
Each data-set recovery task	96KB
IDCAMS load module	80KB
Each recycle task	60KB
Most other DFSMShsm tasks, such as query	50KB

To determine storage required and storage available below the 16 megabyte line, use the following steps:

1. DFSMShsm static storage, system storage, and the DFSMSdss load module together require a total of 2174KB below-the-line storage. This storage is always needed when DFSMShsm is running.
2. To calculate available below-the-line storage after DFSMShsm has started, subtract 2174KB from your region size (for this example, region size is 7148KB).

```
         Region size    7148KB
   Total from step 1  - 2174KB
                        ______

               Total    4974KB
```

3. If, for example, you want to calculate storage that remains available after DFSMShsm migration and backup tasks have been started, multiply the number of migration and backup tasks you are running by the storage required for each task (for this example, use the maximum number of migration and backup tasks, which is 10).

```
Number of migration and backup tasks      10
      Storage required for each task    x 80KB

                               Total     800KB
```

4. Subtract the total storage required for migration and backup tasks from the total remaining storage calculated in step 2 on page 294. This calculates the remaining available storage in which most other DFSMShsm tasks run.

```
                  Total remaining storage    4974KB
Total storage required for migration and backup  -  800KB
                                           ________

                                    Total    4174KB
```

5. In step 3, you can also use tasks other than migration and backup. You can, for instance, total the storage required for all recall tasks you are going to run, and then find the remaining available storage for migration, backup, and other tasks.

Adjusting the size of cell pools

DFSMShsm uses cell pools (the MVS CPOOL function) to allocate virtual storage for frequently used modules and control blocks. Cell pool storage used for control blocks is extendable, while cell pool storage used by modules is not. Using cell pools reduces DFSMShsm CPU usage and improves DFSMShsm performance. The DFSMShsm startup procedure specifies the size (in number of cells) of five cell pools used by DFSMShsm. You can change the default sizes.

If a cell pool runs out of cells, message ARC0019I is issued and DFSMShsm starts using the MVS GETMAIN instruction in place of that cell pool. When this happens, performance is degraded. Using the cell pool number identified in message ARC0019I, increase the size of that cell pool by increasing the corresponding entry in the CELLS keyword in the startup procedure for the DFSMShsm primary address space. You should increase the number of cells by at least the number of cells identified in the message. DFSMShsm must be restarted for the change to take effect.

Table 47 lists the cell pools used by DFSMShsm, the default size, and the maximum recommended size.

Table 47. Default and maximum size for cell pools used by DFSMShsm

Cell pool	Default size	Maximum size
1	200	400
2	100	200
3	100	200

Table 47. Default and maximum size for cell pools used by DFSMShsm (continued)

Cell pool	Default size	Maximum size
4	50	100
5	20	40
Note: If the ARC0019I message is issued, increase the number of cells in the specified cell pool by at least the number of cells identified in the message, even if this is beyond the maximum size indicated in the table above. The maximum size indicates the maximum size anticipated for typical DFSMShsm use. However, running many concurrent DFSMShsm tasks may require an increase beyond the specified maximum value.		

Related reading

- "Specifying the size of cell pools" on page 81
- "DFSMShsm startup procedure" on page 301
- "CELLS (default = (200,100,100,50,20))" on page 305

Chapter 14. DFSMShsm libraries and procedures

The following information discusses DFSMShsm procedure library (PROCLIB) members, parameter library (PARMLIB) members, and procedures (PROCs).

When the DFSMShsm product is installed on your system, SMP processing loads parts of the DFSMShsm product into MVS system libraries during APPLY processing. SMP loads no DFSMShsm parts into the SYS1.PROCLIB and SYS1.PARMLIB. Creating additional parts for SYS1.PROCLIB and SYS1.PARMLIB (or an alternate parameter library) is the responsibility of the MVS system programmer. In fact, SYS1.PROCLIB and SYS1.PARMLIB are system libraries that are intentionally provided for the use of the MVS system programmer.

As an aid to creating the additional parts in SYS1.PROCLIB and in a parameter library, the DFSMShsm product comes with a starter set. The starter set helps installers of DFSMShsm to build SYS1.PROCLIB members and SYS1.PARMLIB members that define a unique DFSMShsm operating environment for your site.

DFSMShsm libraries

This section discusses procedure library (PROCLIB) members and parameter library (PARMLIB) members. To run DFSMShsm, you need a startup procedure in SYS1.PROCLIB and you need a sequence of commands in a parameter library.

Procedure libraries (PROCLIB)

Procedure libraries (PROCLIBs) are data sets that contain JCL procedures and JCL job steps. The procedures include a startup procedure for starting DFSMShsm, a startup procedure for the ABARS secondary address space, and utility jobs for formatting and printing the DFSMShsm logs.

SYS1.PROCLIB is a system library in which the procedures that are included with the DFSMShsm product are placed when you run the starter job. You can create an alternate PROCLIB data set for the startup procedures, the HSMEDIT procedure, and the HSMLOG procedure.

Parameter libraries (PARMLIB)

Parameter libraries (PARMLIBs) are partitioned data sets in which reside a list of commands and directives that MVS reads to determine an operating environment for a program.

If you run the STARTER job provided with the product, the PARMLIB member ARCCMD00 is placed in the SYS1.PARMLIB data set. You can create alternate PARMLIB data sets for ARCCMD*xx* members. There is no requirement to use SYS1.PARMLIB.

When you start DFSMShsm, the PARMLIB pointed to by the HSMPARM DD statement (if there is one) obtains the member ARCCMD00 or an alternate member indicated by the CMD keyword and (if desired) the member ARCSTR00 or an alternate member indicated by the STR keyword. If no HSMPARM DD statement exists, MVS uses concatenated PARMLIB support to obtain members ARCCMD*xx* and ARCSTR*xx*.

Note: If you are using a concatenated parameter library, do not use an HSMPARM DD statement in your startup JCL because it overrides the concatenated PARMLIB support function.

For general information regarding the concatenated PARMLIB support function, refer to the *z/OS MVS Initialization and Tuning Reference*.

Creating alternate DFSMShsm parameter library members

You can create alternate PARMLIB members for different DFSMShsm operating environments.

Note: If you are using concatenated PARMLIB support, refer to the *z/OS MVS System Commands* for information about creating alternate parameter library data sets.

If you use an alternate member, you must:

- Call the alternate member name ARCCMD*xx*, where *xx* is the two characters (numbers or letters) identifying the alternate member name or the alternate startup member ARCSTR*yy* where *yy* is the two characters (numbers or letters) identifying the alternate member name.

Additionally, you must communicate the name of the new PARMLIB member to the MVS operating system. You can either:

- Change the DFSMShsm startup procedure to correspond to the two characters *xx* identifying the alternate member name.

  or
- Use CMD=*xx* or STR=*yy* on the MVS START command for DFSMShsm. Either keyword used this way must be specified on the PROC statement in the DFSMShsm startup procedure.

For example, in Figure 83, the alternate PARMLIB member is named ARCCMD01, so CMD=01 is specified in the startup procedure.

```
 //DFHSM  PROC CMD=01,EMERG=NO,LOGSW=YES,STARTUP=NO,
//       UID=HSM,SIZE=6144K,DDD=50,HOST=1Y
```

Figure 83. Example DFSMShsm Startup Procedure. This JCL directs DFSMShsm to start with the command in PARMLIB member ARCCMD01.

For an example of the entire DFSMShsm startup procedure, see topic “Starter set example” on page 109.

Commands for PARMLIB member ARCCMDxx

Table 48 on page 299 shows:

- The DFSMShsm commands you can specify in the DFSMShsm PARMLIB
- The purpose of the commands
- What information you must re-specify for each startup
- What information you do not have to re-specify for each startup.

Table 48. Commands You Can Specify in the DFSMShsm PARMLIB Member ARCCMDxx

DFSMShsm Command	Purpose	Information You Must Specify Each Startup	Information You Do Not Specify Each Startup
ADDVOL	Adds a volume to DFSMShsm control or defines a space management attribute for a specific volume.	For primary or migration level 1 volumes: • Each volume • Each type of unit • Each type of volume	• Migration level 2 volumes • Backup volumes • Primary volume attributes of a volume you added during an earlier startup • Space management technique of a volume you added during an earlier startup
AUTH	Identifies the user who can issue DFSMShsm-authorized commands. **Note:** This command is only used in a non-FACILITY class environment.	None	*userid* DATABASEAUTHORITY(CONTROL) for the user who can affect the authority of other DFSMShsm users
DEFINE	Defines the control structures within DFSMShsm control.	• Recall pools • Aggregate recovery pools	• Level 2 structure • Backup cycle • Automatic primary space management cycle • Automatic secondary space management cycle • Dump cycle • Dump classes
HOLD	Prevents processing of all or part of DFSMShsm functions.	All parameters	None
ONLYIF	Allows conditional execution of the single command, or group of commands contained within a BEGIN ... END block, immediately following the ONLYIF command.	All parameters	None
PATCH	Changes contents of storage in the address space.	All parameters	None
RELEASE	Releases all or part of the DFSMShsm process that is previously being held using the HOLD command.	All parameters	None
SETMIG	Changes the space management status of data sets, groups of data sets, or all primary volumes.	Migration controls for level qualifiers	Migration controls for data sets or primary volumes
SETSYS	Establishes or changes parameters under which DFSMShsm operates.	All parameters	None
TRAP	Specifies when DFSMShsm should produce a snap dump or an abnormal end dump when a specified error occurs.	All parameters	None

Command sequence for PARMLIB member ARCCMDxx

In the DFSMShsm environment, certain commands must follow a particular sequence to ensure that the command does not malfunction or fail. The following table lists these combinations:

ISSUE THIS COMMAND	BEFORE THIS COMMAND
SETSYS JES2 or JES3	Not Applicable
ADDVOL	DEFINE POOL
DEFINE BACKUP(...	SETSYS NOBACKUP
DEFINE DUMPCLASS	ADDVOL with either the AUTODUMP or the DUMPCLASS parameters
SETSYS MAXRECALLTASKS(*tasks*)	SETSYS TAPEMAXRECALLTASKS(*tasks*)
SETSYS SMALLDATASETPACKING	ADDVOL with SDSP
SETSYS SYSOUT	SETSYS ACTLOGTYPE
SETSYS USERUNITTABLE	ADDVOL
	DEFINE ARPOOL
	DEFINE DUMPCLASS with unit
	SETSYS ARECOVERUNITNAME
	SETSYS BACKUP(TAPE)
	SETSYS CDSVERSIONBACKUP
	SETSYS MIGUNITNAME
	SETSYS RECYCLEOUTPUT
	SETSYS SPILL
	SETSYS TAPEMIGRATION
	SETSYS UNITNAME
	SETSYS TAPEUTILIZATION for any esoteric
SETSYS USERDATASETSERIALIZATION	SETSYS DAYS

Note:

1. If you use the ONLYIF HSMHOST(*hostid*) command with both the DEFINE BACKUP and SETSYS NOBACKUP commands, and assign the commands to different hosts, there is no need for a specific command sequence.
2. SETSYS UNITNAME(*unitname*) should be specified before SETSYS CDSVERSIONBACKUP if you are *not* specifying a unit name as part of the BACKUPDEVICECATEGORY subparameter. If you do not specify SETSYS CDSVERSIONBACKUP BACKUPDEVICECATEGORY UNITNAME(*unitname*), and have not previously specified SETSYS UNITNAME(*unitname*), the default unit is 3590-1.
3. In an HSMplex environment, you should not use the SETSYS EXTENDEDTTOC(Y) command to enable extended TTOCs on any host in the HSMplex until the shared OCDS has been redefined with a record size of 6144 bytes.

Sample libraries (SAMPLIB)

SYS1.SAMPLIB is a system library in which system modification program/extended (SMP/E) logic places the MVS jobs that create rudimentary procedures. If you run the DFSMShsm STARTER job, the DFSMShsm startup procedure is automatically placed in SYS1.PROCLIB and the command sequence (ARCCMD00) is placed in SYS1.PARMLIB. HSMSTPDS places other procedures in a partitioned data set named HSM.SAMPLE.CNTL. You must move the other procedures you want (for example, HSMEDIT and HSMLOG) from HSM.SAMPLE.CNTL to SYS1.PROCLIB.

DFSMShsm procedures

The DFSMShsm procedures and jobs that are provided in SAMPLIB members include:

- A DFSMShsm startup procedure
- An ABARS secondary address space startup procedure
- An installation verification procedure (IVP)
- A functional verification procedure (FVP)
- An HSMLOG procedure to format and print the DFSMShsm log
- An HSMEDIT procedure to print the DFSMShsm edit log

If you are using the starter set, only the HSMLOG procedure and the HSMEDIT procedure must be manually placed in SYS1.PROCLIB. If you are not using the starter set, add the following four procedures to the PROCLIB data set:

1. DFSMShsm startup procedure. This is invoked with an operator START command.
2. ABARS startup procedure. This procedure starts the ABARS secondary address space.
3. HSMLOG procedure. This procedure formats and prints the DFSMShsm log after a log swap has occurred.
4. HSMEDIT procedure. This procedure prints the edit log.

For more information about the DFSMShsm startup procedure for a multiple DFSMShsm-host environment, see “Defining all DFSMShsm hosts in a multiple-host environment” on page 255.

For more information about the startup procedure keywords, see “Startup procedure keywords” on page 302.

DFSMShsm startup procedure

The DFSMShsm startup procedure, shown in Figure 84 on page 307 and shown in the starter set in topic “Starter set example” on page 109 provides the MVS system with DFSMShsm environmental information through the startup procedure keywords and the HSMPARM DD statements, if there are any. You specify these keywords and DD statements to define your processing environment.

Note:

1. If you need to use more startup procedure keywords than can be accommodated by the 100-character PARM limit as detailed in *z/OS MVS JCL Reference* under the EXEC parameter, use the STR=*xx* keyword within the PARM keywords to create a PARMLIB member ARCSTR*xx* to contain the remaining startup parameters.
2. The starter set does not include the RESTART keyword.

When DFSMShsm is started, the MVS operating system reads the DFSMShsm startup procedure and receives information about the DFSMShsm environment.

Startup procedure keywords

The following is a listing and description of the DFSMShsm startup procedure keywords:

CMD (default = 00): Specifies the PARMLIB member that DFSMShsm should start with. The CMD=00 keyword refers to the ARCCMD00 member of a PARMLIB and is discussed in “Parameter libraries (PARMLIB)” on page 297. Throughout the DFSMShsm library the term ARCCMD*xx* is used to discuss the PARMLIB member. You can create more than one ARCCMD*xx* member and designate a different number for each PARMLIB member you create by substituting a number for *xx* (the last two characters of the ARCCMD*xx* member).

EMERG (default = NO): Specifies whether DFSMShsm starts processing immediately. The EMERG=NO keyword allows DFSMShsm to begin functioning as soon as it is started. The EMERG=YES keyword allows DFSMShsm to start, but does not allow DFSMShsm to perform any functions until a SETSYS NOEMERGENCY command is issued.

LOGSW (default = NO): Specifies whether to swap the DFSMShsm log data sets automatically at startup. Because the problem determination aid (PDA) logs are automatically swapped at startup, you should specify LOGSW=YES to synchronize the DFSMShsm logs with the PDA logs. If you specify LOGSW=YES, you must also change your JCL disposition (DISP) to DISP=OLD for the LOGX and LOGY data sets.

For a discussion of the DFSMShsm log data sets, see “DFSMShsm log data set” on page 45.

STARTUP (default = NO): Specifies whether DFSMShsm displays startup messages at the operator console.

UID (default = HSM): Specifies the DFSMShsm authorized-user identification (UID) in 1 to 7 characters. You must use this UID as the first qualifier of the data set name of the SDSP data sets. In the DFSMShsm starter set sample jobs (see Chapter 6, “DFSMShsm starter set,” on page 101), the UID is the prefix name for the migrated and backed up data sets. The UID is also the first qualifier of the DFSMShsm log in the DFSMShsm started procedure in topic “Starter set example” on page 109. For the starter set, HSM is the UID.

Note: Changing the UID parameter after your environment has been set up requires coordination, because DFSMShsm expects the UID to be the high-level qualifier for SDSP data sets. If you change the UID, you must also change the high-level qualifier of existing SDSP data sets to match the new UID. Although the UID also appears as the high-level qualifier of tape data sets, DFSMSrmm and most other tape management systems allow the high-level qualifier to be different.

HOSTMODE (default = MAIN): Specifies how this instance of DFSMShsm is related to various functions of DFSMShsm.

HOSTMODE=MAIN specifies that this DFSMShsm:

- Processes implicit requests, like recalls and deleting migrated data sets, from user address spaces
- Processes explicit commands from TSO, like HSENDCMD and HBACKDS
- Manages ABARS secondary address spaces
- Allows MODIFY commands from a console

- Can run an automatic backup, dump, and space management

Within a z/OS image, only one DFSMShsm can operate in this mode and any other DFSMShsm host in that image must have HOSTMODE=AUX.

HOSTMODE=AUX specifies that this DFSMShsm:
- Allows MODIFY commands from a console
- Can run automatic backup, dump, or space management

Within a z/OS image, zero or more DFSMShsm hosts can operate in this mode.

If HOSTMODE is not specified, the default is MAIN.

SIZE: Specifies the region size in K bytes (K=1024) that DFSMShsm is running under. You should specify 0M for SIZE which directs DFSMShsm to allocate the largest possible region size. For more detailed information about DFSMShsm storage requirements, see “DFSMShsm address spaces” on page 293.

Note: The DFSMShsm region size requirements depend on the type and number of concurrent data movement tasks.

DDD: Specifies the number of dynamically allocated resources that can be held in anticipation of reuse. This value is used in the DYNAMNBR parameter of the EXEC statement. Refer to *z/OS MVS JCL Reference* for further explanation of the DYNAMNBR parameter.

HOST=x: Identifies this DFSMShsm host in an HSMplex. The HOST=*x* keyword specifies a unique identifier for each instance of DFSMShsm. For *x*, substitute the host identification as an upper-case alphabetic character from A to Z, a digit from 0 to 9, or the character @, #, or $.

Note:
1. The earlier definition of the HOST= keyword allowed an optional second character in the value. The function of that second character is now specified by the PRIMARY= keyword. The second character, if specified, is considered only if the PRIMARY= keyword is not specified.
2. DFSMShsm supports the use of @, #, and $ as host identifier characters that may show up in dataset names and TSO output. You are not required to use them and, (especially if you are a non-US customer) you may choose to limit yourself to characters A-Z, and 0-9, reducing the total number of host address spaces per HSMplex to 36 instead of 39.

PRIMARY (default = YES): Specifies whether this DFSMShsm host is the primary host within its HSMplex, and thus performs the backup and dump level functions as part of automatic processing. Automatic primary space management and automatic secondary management can be performed on any DFSMShsm host.

If you do not specify the PRIMARY= keyword:
- If the HOST= keyword value has a second character Y, this DFSMShsm host is the primary host
- If the HOST= keyword value has a second character N, this host is not the primary host
- If the HOST= keyword has no valid second character, this host is the primary host

RESTART: Specifies that DFSMShsm should be automatically restarted for all DFSMShsm abnormal ends. The RESTART='(*a*, *b*)' keyword specifies that DFSMShsm should be automatically restarted for all DFSMShsm abnormal ends. *a* specifies the name of the procedure to be started and *b* specifies any additional keywords or parameters to be passed to the procedure. For example, if the DFSMShsm procedure DFHSM01 is started for all automatic restarts, and EMERG is set to YES, then the RESTART keyword would be specified as: RESTART='(DFHSM01.HSM,EMERG=YES)'. Note that in this example HSM can be used by the operator as an identifier for DFHSM01.

Note: If you are accessing your CDSs in RLS mode, use the RESTART keyword so that DFSMShsm automatically restarts after shutting down due to SMS VSAM server error.

For a detailed example of using the RESTART keyword to restart DFSMShsm after an abnormal end, see "Using the RESTART keyword to automatically restart DFSMShsm after an abnormal end" on page 306.

CDSQ: Specifies that DFSMShsm serializes its control data sets with a global enqueue product (GRS for example) instead of serializing with volume reserves.

When you specify YES for this parameter, DFSMShsm serializes the use of the control data sets (between multiple z/OS images) with a global (SYSTEMS) exclusive enqueue and still allows multiple tasks within a single z/OS image to access the control data sets concurrently. All DFSMShsm hosts within an HSMplex must use the same serialization technique.

If you specify CDSQ=NO (without CDSSHR=RLS), the only allowable HOSTMODE for any DFSMShsm host within the HSMplex is MAIN.

For more information about serializing CDSs with the CDSQ keyword, see "Serialization of control data sets with global resource serialization" on page 262.

CDSR: Specifies that DFSMShsm serializes its control data sets with volume reserves.

If you have installed a global resource serialization (GRS) product, you can serialize your CDSs with GRS.

When you specify YES for this parameter, DFSMShsm serializes the use of the control data sets with a shared ENQ/RESERVE.

All the hosts in an HSMplex must implement the same serialization technique.

When a serialization technique has not been specified, the default serialization technique depends on the following specified HOSTMODE:
- If HOSTMODE=MAIN, DFSMShsm assumes CDSR=YES
- If HOSTMODE=AUX, DFSMShsm indicates an error with message ARC0006I

For more information about serializing CDSs with the CDSR keyword, see "Serialization of control data sets with global resource serialization" on page 262.

CDSSHR: Specifies that the DFSMShsm being started will run in a particular multiple-image or single-image environment.

Because this keyword is not normally specified, it has no default value. Its main uses are for testing and for merging multiple CDSs.

When you specify NO for this keyword, DFSMShsm does no multiple-host serialization; no other system should be concurrently processing this set of CDSs. The HOSTMODE of this DFSMShsm can only be MAIN. For performance reasons, specify NO in a single image environment with no auxiliary hosts and where the index of the MCDS is on a DASD device that is configured as shared.

When you specify YES for this keyword, DFSMShsm does multiple-host serialization of the type requested by the CDSQ and CDSR keywords.

When you specify RLS for this keyword, DFSMShsm performs multiple-host serialization using record level sharing (RLS). When RLS is specified, the CDSs are accessed in RLS mode and any values specified for CDSQ and CDSR are ignored.

If you do not specify the CDSSHR keyword in the startup procedure, DFSMShsmperforms multiple-host serialization if the index component of the MCDS resides on a DASD volume that has been SYSGENed as SHARED or SHAREDUP.

CELLS (default = (200,100,100,50,20)): DFSMShsm uses the cell-pool (CPOOL) function of MVS to obtain and manage virtual storage in its address space for the dynamically obtained storage for certain high-usage modules, and for data areas DFSMShsm frequently gets and frees. The CELLS parameter provides the cell sizes for five cell pools.

For more information about DFSMShsm storage, see “Adjusting the size of cell pools” on page 295.

PDA (default = YES): Specifies that problem determination aid (PDA) tracing begins before the SETSYS PDA command has been processed. When you specify YES for this parameter, the DFSMShsm problem determination aid facility begins its tracing functions at the beginning of startup processing instead of waiting for a SETSYS PDA command or instead of waiting for DFSMShsm to complete its initialization.

For more information about the PDA trace function, see “DFSMShsm problem determination aid facility” on page 41.

RNAMEDSN (default = NO): Specifies whether to use a new serialization method so that there is no longer interference between HSMplexes that are contained within a single GRSplex. When you specify YES for this parameter, you are invoking the new method of serialization, which uses the data set name of the CDSs and the journal.

For more information about the GRSplex serialization function, see Chapter 12, “DFSMShsm in a sysplex environment,” on page 279.

STR: Specifies a PARMLIB member containing DFSMShsm startup parameters, which are logically concatenated with any remaining parameters specified on the EXEC statement. The value for the STR keyword must be two characters, but it need not be the same as the value for the CMD keyword.

As with member ARCCMD*xx*, if you are not using MVS concatenated PARMLIB support, member ARCSTR*xx* must be in the data set specified with DD statement

HSMPARM in the startup procedure. If you are using MVS concatenated PARMLIB support, members ARCCMD*xx* and ARCSTR*xx* need not be in the same PARMLIB data set.

No other keywords need be specified with PARM= on the EXEC statement, but note that no substitution of symbolic parameters occurs in member ARCSTR*xx*. Thus parameters specified on the START command are limited to symbolic parameters specified on the PROC statement.

Each record in member ARCSTR*xx* contains one or more startup keywords, separated by commas. There is no explicit continuation character defined. DFSMShsm assumes that the last eight characters (73 – 80) in each record are a sequence number field, and does not scan that field. Keywords can be specified in any order. If the same keyword is specified more than once, the last instance is the one that is used.

If the first nonblank characters in a record are "/*" DFSMShsm considers the record a comment and ignores it.

If a keyword is specified both with PARM= and in the ARCSTR*xx* member, the specification in PARM= overrides that in the member.

If member ARCSTR*xx* exists, DFSMShsm reads each record and processes its parameters as if they had been specified using PARM=. Then the parameters (if any) specified with PARM= are processed.

Neither an empty member nor the absence of the STR= keyword is considered an error.

You can use the STR keyword for at least two purposes:
- To allow specifying more startup parameters than can be accommodated in the PARM= field
- To split keywords between host-unique ones in the PARM= field, for example;
  ```
  PARM=('CMD=&CMD', 'HOST=&HOST', 'STR=&STR')
  ```

Using the RESTART keyword to automatically restart DFSMShsm after an abnormal end: If you are running DFSMShsm in an MVS/ESA environment, you can use the RESTART keyword in the DFSMShsm startup procedure to automatically restart DFSMShsm after an abnormal termination.

When you specify the RESTART keyword in the DFSMShsm start procedure, no operator intervention is required to automatically restart DFSMShsm. The protocol for the RESTART keyword is: RESTART=(A,B), where A is required and is the name of the DFSMShsm procedure to be started, and B is optional and specifies any additional parameters to be passed to the procedure.

Figure 84 on page 307 is an example of the DFSMShsm startup procedure. Notice that the EMERG keyword NO allows all DFSMShsm functions and that the RESTART keyword restarts DFSMShsm automatically. However, when DFSMShsm is restarted the status of the EMERGENCY keyword in the restarted procedure is YES. When EMERG=YES, DFSMShsm does not allow any functions to start.

```
//*********************************************************************/
//* EXAMPLE DFSMSHSM STARTUP PROCEDURE THAT SPECIFIES THE RESTART      */
//* KEYWORD TO RESTART DFSMSHSM WITH A DIFFERENT STATUS FOR THE EMERG  */
//* KEYWORD.                                                           */
//*********************************************************************/
//*
//DFSMSHSM   PROC CMD=00,         USE PARMLIB MEMBER ARCCMD00
//            LOGSW=YES,          SWITCH LOGS AT STARTUP
//            STARTUP=YES,        STARTUP INFO PRINTED AT STARTUP
//            UID=HSM,            DFSMSHSM-AUTHORIZED USER ID
//            PDA=YES,            BEGIN PDA TRACING AT STARTUP
//            SIZE=0M,            REGION SIZE FOR DFSMSHSM
//            DDD=50,             MAX DYNAMICALLY ALLOCATED DATA SETS
//            HOST=?HOSTID,       PROC.UNIT ID AND LEVEL FUNCTIONS
//            PRIMARY=?PRIMARY,   LEVEL FUNCTIONS
//            RESTART='(DFHSM00,EMERG=YES)'    RESTART INFORMATION
//DFSMSHSM  EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//          PARM=('LOGSW=&LOGSW','CMD=&CMD','UID=&UID',
//          'HOST=&HOSTID','PRIMARY=&PRIMARY',
//          'STARTUP=&STARTUP','PDA=&PDA','RESTART=&RESTART')
//*********************************************************************/
//*     HSMPARM DD must be deleted from the JCL or made into a         */
//*         a comment to use Concatenated PARMLIB support.             */
//*********************************************************************/
//HSMPARM  DD DSN=SYS1.PARMLIB,DISP=SHR
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//SYSPRINT DD SYSOUT=A,FREE=CLOSE
//SYSUDUMP DD SYSOUT=A
//MIGCAT   DD DSN=&UID...MCDS,DISP=SHR
//BAKCAT   DD DSN=&UID...BCDS,DISP=SHR
//OFFCAT   DD DSN=&UID...OCDS,DISP=SHR
//JOURNAL  DD DSN=&UID...JRNL,DISP=SHR
//ARCLOGX  DD DSN=&UID...HSMLOGX1,DISP=OLD
//ARCLOGY  DD DSN=&UID...HSMLOGY1,DISP=OLD
//ARCPDOX  DD DSN=&UID...HSMPDOX1,DISP=OLD
//ARCPDOY  DD DSN=&UID...HSMPDOY1,DISP=OLD
//*
```

Figure 84. Example of Automatically Restarting DFSMShsm. The status of the keyword EMERG is changed when DFSMShsm automatically restarts.

Figure 85 on page 308 shows an alternate way of obtaining the same results; restart DFSMShsm using a different startup procedure with a different name.

Upon an abnormal termination of DFHSM00, DFHSM05, shown in Figure 86 on page 308, is started with an identifier of HSM. The procedure DFHSM05 does not specify a RESTART keyword and specifies EMERG=YES. If DFSMShsm ends with an abnormal termination, DFSMShsm does not automatically restart.

Rule: Any alternate procedures (DFHSM05, for example) must have an entry in the RACF started-procedures table and must be associated with the DFSMShsm user ID for RACF.

```
//********************************************************************/
//* EXAMPLE DFSMSHSM STARTUP PROCEDURE THAT RESTARTS DFSMSHSM WITH A   */
//* DIFFERENT STARTUP PROCEDURE (DFHSM05) AFTER AN ABEND.              */
//********************************************************************/
//*
//DFSMSHSM   PROC CMD=00,         USE PARMLIB MEMBER ARCCMD00
//            LOGSW=YES,          SWITCH LOGS AT STARTUP
//            STARTUP=YES,        STARTUP INFO PRINTED AT STARTUP
//            UID=HSM,            DFSMSHSM-AUTHORIZED USER ID
//            PDA=YES,            BEGIN PDA TRACING AT STARTUP
//            SIZE=0M,            REGION SIZE FOR DFSMSHSM
//            DDD=50,             MAX DYNAMICALLY ALLOCATED DATA SETS
//            HOST=?HOSTID,       PROC.UNIT ID AND LEVEL FUNCTIONS
//            PRIMARY=?PRIMARY,   LEVEL FUNCTIONS
//            RESTART='(DFHSM05,EMERG=YES)'    RESTART INFORMATION
//DFSMSHSM  EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//          PARM=('LOGSW=&LOGSW','CMD=&CMD','UID=&UID',
//          'HOST=&HOSTID','PRIMARY=&PRIMARY',
//          'STARTUP=&STARTUP','PDA=&PDA','RESTART=&RESTART')
//********************************************************************/
//*     HSMPARM DD must be deleted from the JCL or made into a         */
//*         a comment to use Concatenated PARMLIB support.             */
//********************************************************************/
//HSMPARM  DD DSN=SYS1.PARMLIB,DISP=SHR
//MSYSOUT  DD SYSOUT=A
//MSYSIN   DD DUMMY
//SYSPRINT DD SYSOUT=A,FREE=CLOSE
//SYSUDUMP DD SYSOUT=A
//MIGCAT   DD DSN=&UID...MCDS,DISP=SHR
//BAKCAT   DD DSN=&UID...BCDS,DISP=SHR
//OFFCAT   DD DSN=&UID...OCDS,DISP=SHR
//JOURNAL  DD DSN=&UID...JRNL,DISP=SHR
//ARCLOGX  DD DSN=&UID...HSMLOGX1,DISP=OLD
//ARCLOGY  DD DSN=&UID...HSMLOGY1,DISP=OLD
//ARCPDOX  DD DSN=&UID...HSMPDOX1,DISP=OLD
//ARCPDOY  DD DSN=&UID...HSMPDOY1,DISP=OLD
//*
```

Figure 85. Example of Automatically Restarting DFSMShsm with a Different Procedure. If an abnormal end occurs, the startup procedure calls a different startup procedure (DFHSM05), shown in Figure 86.

```
//********************************************************************/
//* EXAMPLE DFHSM05 STARTUP PROCEDURE THAT IS CALLED FROM THE          */
//* PRECEDING STARTUP PROCEDURE FOR UNEXPECTED DFSMSHSM ABENDS.        */
//********************************************************************/
//*
//DFSMSHSM   PROC CMD=00,         USE PARMLIB MEMBER ARCCMD00
//            LOGSW=YES,          SWITCH LOGS AT STARTUP
//            STARTUP=YES,        STARTUP INFO PRINTED AT STARTUP
//            UID=HSM,            DFSMSHSM-AUTHORIZED USER ID
//            PDA=YES,            BEGIN PDA TRACING AT STARTUP
//            SIZE=0M,            REGION SIZE FOR DFSMSHSM
//            DDD=50,             MAX DYNAMICALLY ALLOCATED DATA SETS
//            HOST=?HOSTID,       PROC.UNIT ID AND LEVEL FUNCTIONS
//            PRIMARY=?PRIMARY    LEVEL FUNCTIONS
//DFSMSHSM  EXEC PGM=ARCCTL,DYNAMNBR=&DDD,REGION=&SIZE,TIME=1440,
//          PARM=('LOGSW=&LOGSW','CMD=&CMD','UID=&UID',
//          'HOST=&HOSTID','PRIMARY=&PRIMARY',
//          'STARTUP=&STARTUP','PDA=&PDA'
//                        .
//                        .
//                        .
//                        .
```

Figure 86. Example of Alternate Startup Procedure

As you can see, the RESTART keyword not only allows you to tell DFSMShsm to restart itself, but allows you to modify the manner in which DFSMShsm is restarted when an abnormal termination occurs. Any keyword that can be specified in the MVS start command can be specified in the RESTART keyword as part of parameter B.

Startup procedure DD statements

If you are not using concatenated PARMLIB support, this section discusses the required DD statements for the HSMPARM statement of the DFSMShsm startup procedure. The DD statement names must appear as they are shown in the DFSMShsm starter set topic “Starter set example” on page 109. The high-level qualifier (HLQ) for these required DD statements does not have to be UID and the control data sets are not required to share the same high-level qualifier.

Required DD Name	Description
HSMPARM DD DSN=HLQ...PARMLIB, DISP=SHR	This DD statement identifies the PARMLIB member that contains commands and directives to establish a DFSMShsm operating environment. The FREE=CLOSE parameter on the DD statement should NOT be used, as DFSMShsm will automatically deallocate the parmlib dataset.
MSYSOUT DD SYSOUT=A	This DD statement identifies a system data set that provides DFSMShsm with the messages issued by the terminal monitor program (TMP) and with messages issued when dynamic memory allocation takes place.
MSYSIN DD DUMMY	This DD statement identifies a system data set that provides DFSMShsm with a DUMMY SYSIN data set for DFSMShsm support of TSO processing.
SYSPRINT DD SYSOUT=A, FREE=CLOSE	This DD statement identifies the output destination for SYSPRINT requests.
SYSUDUMP DD SYSOUT=A	This DD statement identifies the SYSOUT class for SYSUDUMPs.
MIGCAT DD DSN=HLQ...MCDS, DISP=SHR	This DD statement identifies the migration control data set to DFSMShsm.
BAKCAT DD DSN=HLQ...BCDS, DISP=SHR	This DD statement identifies the backup control data set to DFSMShsm.
OFFCAT DD DSN=HLQ...OCDS, DISP=SHR	This DD statement identifies the offline control data set to DFSMShsm.
JOURNAL DD DSN=HLQ...JRNL, DISP=SHR	This DD statement identifies the journal data set to DFSMShsm.
ARCLOGX DD DSN=HLQ...HSMLOGX1, DISP=OLD	This DD statement identifies the LOGX (DFSMShsm log) data set to DFSMShsm. If you specify LOGSW=YES, specify DISP=OLD.
ARCLOGY DD DSN=HLQ...HSMLOGY1, DISP=OLD	This DD statement identifies the LOGY (DFSMShsm log) data set to DFSMShsm. If you specify LOGSW=YES, specify DISP=OLD.
ARCPDOX DD DSN=HLQ...HSMPDOX, DISP=OLD	This DD statement identifies the PDOX (PDA trace) data set to DFSMShsm.

Required DD Name	Description
ARCPDOY DD DSN=HLQ...HSMPDOY, DISP=OLD	This DD statement identifies the PDOY (PDA trace) data set to DFSMShsm.

Using DD statement AMP parameters to override DFSMShsm default values: Default values specified in the ACB are used by DFSMShsm when opening a CDS. The default values specified in the ACB are: STRNO=40, BUFND=41, and BUFNI=60. You can override the default values by specifying AMP parameters in a DD statement for each CDS.

In general, you should not override the default values unless the values are increased. For example, if VSAM dynamic string addition is causing a problem in your operating environment, you can increase the number of strings (STRNO) used for concurrent access by specifying an AMP parameter in the DD statement.

For an example of using DD statement AMP parameters to override the default values specified in the ACB, see "Example of using a DD statement to override default values stored in the ACB."

Note:

1. When using the AMP parameter to increase the number of strings, the value specified for STRNO should be determined by the number of concurrent requests you expect to process for a given CDS. The maximum value is 255 strings. However, the region size must be large enough to support the increased size of extended private storage used for VSAM control blocks when increasing the STRNO value.
2. When the STRNO value is changed, the BUFND and BUFNI values might also need to be changed. The BUFND value should be 1 greater than the STRNO value and a large BUFNI value can increase performance.

For more information, see the topics about processing VSAM data sets and optimizing VSAM performance in *z/OS DFSMS Using Data Sets*.

Example of using a DD statement to override default values stored in the ACB: The following example overrides the default values stored in the ACB by specifying AMP parameters in the DD statement; where *nnn* is the desired value

```
//MIGCAT DD DSN=&UID..MCDS,DISP=SHR, AMP=('STRNO=nnn','BUFND=nnn','BUFNI=nnn')
```

ABARS secondary address space startup procedure

The ABARS startup procedure, shown in Figure 87 on page 311, and in the starter set topic "Starter set example" on page 109, provides the MVS operating system with environmental information about the ABARS secondary address space.

When DFSMShsm is started, the MVS operating system reads the ABARS startup procedure to get information about the ABARS secondary address space.

```
//**********************************************************************/
//*          ABARS SECONDARY ADDRESS SPACE STARTUP PROCEDURE          */
//**********************************************************************/
//*
//DFHSMABR PROC
//DFHSMABR EXEC PGM=ARCWCTL,REGION=0M
//SYSUDUMP DD SYSOUT=A
//MSYSIN DD DUMMY
//MSYSOUT DD DUMMY
//*
```

Figure 87. Example of an Aggregate Backup and Recovery Startup Procedure. This procedure starts the ABARS secondary address space.

DFSMShsm installation verification procedure (IVP) startup procedure

The installation verification procedure (IVP), described in Chapter 2, "Installation verification procedure," on page 7, is a procedure that exercises and tests the SMP/E installation of the DFSMShsm product.

DFSMShsm functional verification procedure (FVP)

The functional verification procedure (FVP), described in detail in Chapter 8, "Functional verification procedure," on page 153, is a procedure that can be used to exercise and test the functions of DFSMShsm.

HSMLOG procedure

The HSMLOG procedure, shown in Figure 88, is a procedure that formats and prints the DFSMShsm log and places selected information in the edit log.

```
//HSMLOG   JOB JOBPARM
//*
//**********************************************************************/
//* THIS SAMPLE JOB PRINTS THE DFSMSHSM LOG.  REPLACE THE UID VARIABLE */
//* WITH THE DFSMSHSM-AUTHORIZED USER ID ( 1 TO 7 CHARACTERS).         */
//**********************************************************************/
//*
//PRINTLOG EXEC PGM=ARCPRLOG
//ARCPRINT DD SYSOUT=*
//ARCLOG DD DSN=UID.HSMLOGY1,DISP=OLD
//ARCEDIT DD DSN=UID.EDITLOG,DISP=OLD
//*
//EMPTYLOG EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSUT2 DD DSN=UID.HSMLOGY1,DISP=OLD
//SYSUT1 DD DUMMY,DCB=(UID.HSMLOGY1)
/*
```

Figure 88. Example of the HSMLOG Procedure

Note: Do not compress the log data set that is used as input to the ARCPRLOG program. The log data set is created with RECFM=F, but is opened by ARCPRLOG for update with RECFM=U, which is not allowed for compressed data sets.

HSMEDIT procedure

The HSMEDIT procedure, shown in Figure 89 on page 312 and in the starter set in "HSMEDIT" on page 142, is a procedure that prints the edit log.

```
//EDITLOG  JOB JOBPARM
//*
//*********************************************************************/
//* THIS SAMPLE JOB PRINTS THE DFSMSHSM EDIT LOG.  REPLACE THE UID     */
//* VARIABLE WITH THE DFSMSHSM-AUTHORIZED USER ID (1 TO 7 CHARACTERS). */
//*********************************************************************/
//*
//EDITLOG EXEC PGM=ARCPEDIT
//ARCPRINT DD SYSOUT=*
//ARCLOG DD DSN=UID.EDITLOG,DISP=SHR
/*
```

Figure 89. Example of the HSMEDIT Procedure

Note: To send the output to a data set (see Figure 90), change ARCPRINT to:

```
//ARCPRINT DD DSN=uid.EDITOUT,DISP=(NEW,CATLG),UNIT=unitname,
//     VOL=SER=volser,SPACE=spaceinfo,
//     DCB=(RECFM=FBA,LRECL=133,BLKSIZE=26600)
```

Figure 90. Example of a Change to ARCPRINT

Chapter 15. User application interfaces

This information is intended to help customers use application programs to gather data from DFSMShsm. This section documents General-use Programming Interfaces and Associated Guidance Information provided by DFSMShsm.

The following information focuses on collecting system data through a DFSMShsm-related program called ARCUTIL, also invoked through a DFSMSdfp data collection command called DCOLLECT. The following discussion is divided into five parts:
- Data collection
- Capacity planning
- Data collection invocation
- Data collection exit support
- DFSMSrmm Reporting

Data collection

The collection of pertinent system data allows storage administrators and capacity planners to effectively and efficiently plan for and manage their systems. ARCUTIL, the DFSMShsm data collection interface, captures a snapshot copy of DFSMShsm-specific information in a physical sequential data set. The name of the physical sequential data set is the collection data set. The collection data set includes the following DFSMShsm-specific records:
- Migrated data set information record
- Backup version information record
- Tape capacity planning record
- DASD capacity planning record

Note: If you use the IDCAMS DCOLLECT command to invoke ARCUTIL, the collection data set can contain other records as well, depending on the parameters specified for DCOLLECT.

z/OS DFSMS Access Method Services Commands gives a description of the fields within these four (and other) records, and "Invoking the DFSMShsm data collection interface" on page 320 discusses registers and parameters used with this interface. By using your own program to select the fields you want, you can generate any kind of report in any format, or you can use a preprogrammed report generator (for example, NaviQuest) for your system reports.

Planning to use data collection

Before you access the collection data set, some planning decisions must be made. You will need to decide on:
- The method you use to collect data
- The method you use to create reports
- The types of reports you want

The following will help you to make your choices.

Choosing a data collection method

Figure 91 on page 317 shows three job control language (JCL) entry points from which the ARCUTIL load module can be accessed. Because ARCUTIL first tries to access the CDS in RLS mode, you can ignore message IEC161I 009 if you have not specified record level sharing. This section discusses those entry points. You can choose any of three methods for collecting system data:

- Invocation of the IDCAMS DCOLLECT command.

  DCOLLECT is a command in Access Method Services (AMS), available in MVS/DFP™ Version 3 Release 2 or later. By using IDCAMS DCOLLECT, you can create reports including not only DFSMShsm-specific but also DFSMSdfp-specific information.

  You can use the following parameters:

  Parameter
  : **Data Collected**

  MIGRATEDATA
  : Migrated data set information

  BACKUPDATA
  : Backup version information

  CAPPLANDATA
  : Tape capacity and DASD capacity planning

- Direct invocation of ARCUTIL using JCL.

  You can access the ARCUTIL module with JCL and receive DFSMShsm-specific data.

- Invocation of a user-written program.

  A user-written program may request DFSMShsm to create the collection data set. By writing your own program, you can generate custom reports for your environment without using the IDCAMS DCOLLECT command. For an example of a user-written program, see Figure 96 on page 323.

Choosing a report creation method

Figure 91 on page 317 shows two reporting paths leading to reports. This section discusses some possibilities for creating reports.

- You can access the records of the data collection data set through your own program. A user-written program can produce a report that addresses the specific needs of an installation. This approach is more flexible, but requires some additional programming to produce the custom reports.

  See the DCOLREXX sample program written in the TSO/E REXX programming language as an example for processing these records. Chapter 7, “DFSMShsm sample tools,” on page 151 describes the way to access this program.

- You can use simple reports that have been predefined and are available through the NaviQuest product. NaviQuest provides support for DCOLLECT data by including a starter set of Log, Summary, and Parameter Tables and Views. For additional information, refer to the NaviQuest appendix in the *z/OS DFSMSdfp Storage Administration* .
- The DFSORT™ product includes ICETOOL, a DFSORT utility that makes it easy for you to create reports. A set of illustrative examples of analyzing data created by DFSMShsm, DFSMSrmm, DCOLLECT, and SMF are included with the DFSORT R13 product. Refer to *z/OS DFSORT Application Programming Guide* under “Storage Administrator Examples” for more information.

- The DFSMSrmm Report Generator can be used with utilities like DFSORT's ICETOOL to create customized reports. You can create DFSMShsm report definitions, save reporting jobs, and submit reporting jobs using the DFSMSrmm Report Generator.

  For information on FSR and DCOLLECT Records see Running New Reports with Report Generator in Chapter 14, Obtaining Information from DFSMShsm of *z/OS DFSMShsm Storage Administration*.

  For more detailed information of how the DFSMSrmm Report Generator works see Chapter 2, Using the DFSMSrmm Report Generator in *z/OS DFSMSrmm Reporting*.

  To add and change report types so DCOLLECT users can use DFSMSrmm Report Generator see Figure 28. Adding a Report Type Using the Add a Report Type Panel and Figure 32. Changing a Report Type Using the Change a Report Type Panel in Chapter 2, Using the DFSMSrmm Report Generator of *z/OS DFSMSrmm Reporting*.

```
Panel  Help
------------------------------------------------------------------------------
EDGPG022           DFSMSrmm Report Generation - DCOLLECT
Command ===>

Enter or change the skeleton variables for the generated JCL:

Input data set  . . . . 'DFRMM1.DCOLLECT'

Date format . . . . . . ISO
    (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 . . . : IDCDOUT
                                      and macro keyword . . : TYPE=V
  Enter END command to start the report generation or CANCEL
```

  See Running a Report Generator in Chapter 2. Using the DFSMSrmm Report Generator in *z/OS DFSMSrmm Reporting* for Figure 3. Select the Input Data Set in the Product Library Using the DFSMSrmm Report Definitions Search Panel and Figure 6. Running Your Report Using the DFSMSrmm Report Definitions Panel in Running a Report Generator Report. Also, see Figure 27. DFSMSrmm Report Types Panel in Working with Report Types in the same chapter.

  The report definitions and report types specify the format and contents of reports, the input files for the reports, and the tools used to create the reports. To use or modify a report, you work with report definitions. Create new report definitions for reports that are required by your users. Store the report definitions in the installation library to make the reports available to all your users from the installation library. To create a new report that uses input data other than the DFSMShsm files, you work with report types.

  You store report definitions, report types, and the reporting tools in three separate libraries.

  – The product library which contains predefined report definitions, report types, and reporting tools.
  – The installation library which contains any versions that your installation has modified or created.
  – The user library where any new or modified versions are stored.

The DFSMSrmm Report Generator supports a list of up to five assembler macros to map the data in records to be used for reporting. For each macro you can specify one or more keywords with values to be used with the macro name at assembly time. The macros are assembled and the assembler listing is used to extract field information. The offset for each field, its characteristics and length are saved to be used for selection by the dialog user.

ARCUTILP provides keyword options so that only a single type of record can be mapped to simplify use under the report generator:

```
ARCUTILP IDCDOUT=YES/NO,TYPE=ALL/M/B/C/T
```

ISPF skeletons generate extract steps for DCOLLECT and update SMF extracts for use with DFSMShsm FSR report types. You need to tailor the skeleton to perform processing based on the JCL and control statements required for your selected data and reporting utility.

The DFSMShsm supplied skeletons ARCGFSRC and ARCGWFSC are used by the generator to convert FSR and WWFSR records to FSR2 and WFSR2 records respectively. For details about reporting with DFSMShsm and DCOLLECT data, see the DFSMShsm section of *z/OS DFSMSdfp Storage Administration*.

Choosing the type of report you want

Applications can collect data periodically to provide reporting for the following functions:

- Capacity planning
- Billing and cost accounting
- Storage and space administration

Figure 91 on page 317 shows an overview of the DFSMShsm data collection interface. From the figure, you can see the relationships of your choices to the DCOLLECT data path.

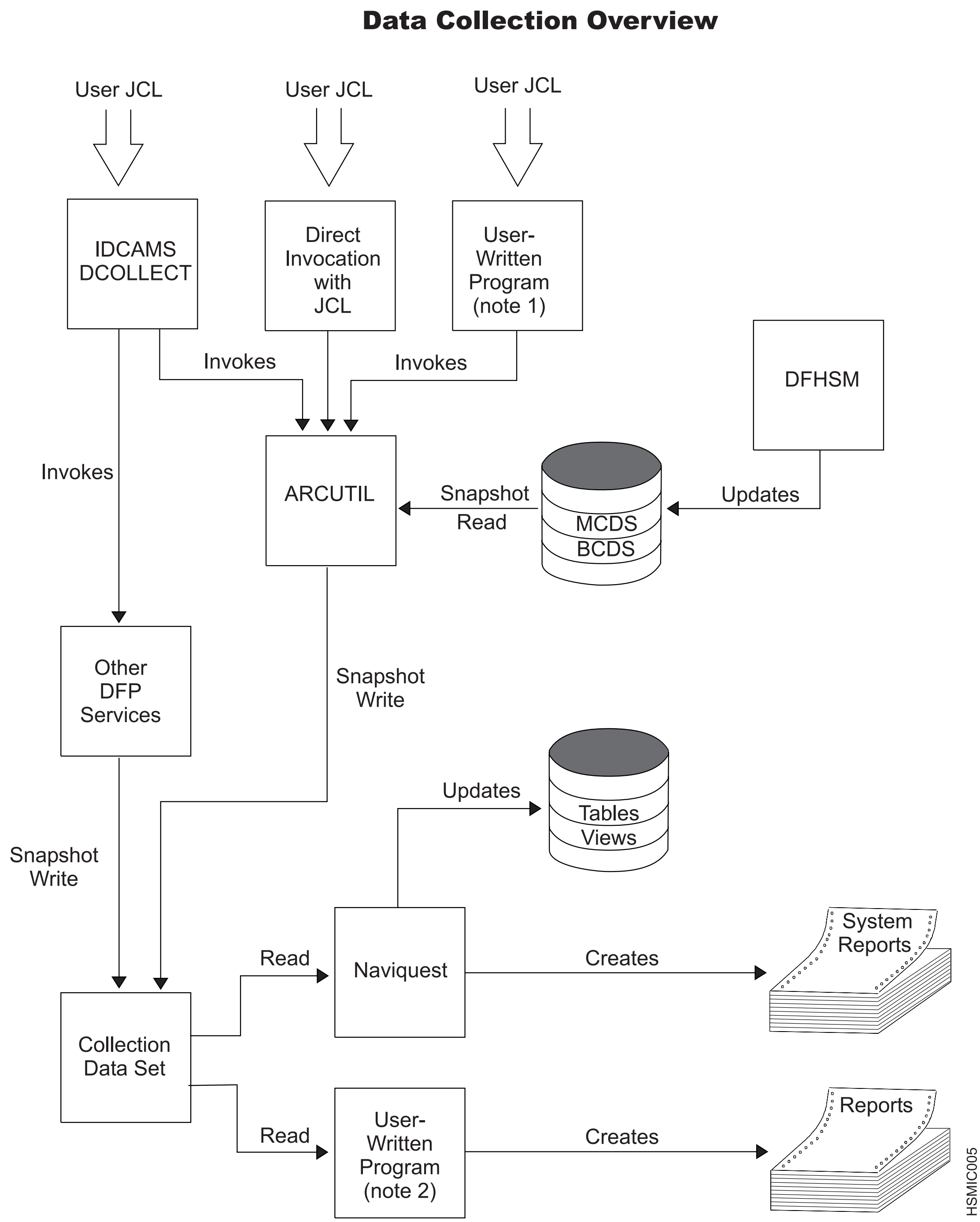


Figure 91. Data Collection Overview

Note:

1. The ARCUTIL load module can be accessed by a user-written program (as opposed to using IDCAMS). A sample program that will access ARCUTIL is described in topic Figure 96 on page 323.
2. Custom reports for your installation can be produced by writing your own program to format the data available in the collection data set.

The data collection environment

This section provides information about the data sets required for data collection, describes the data collection records, explains invocation parameters, shows sample data collection programs, and defines return codes and reason codes.

Data sets used for data collection

Data collection by ARCUTIL uses the following data sets:

- The migration control data set (MCDS)
- The backup control data set (BCDS)
- The snap processing data set
- The collection data set

Note: ARCUTIL does not support data sets allocated with any of the following three dynamic allocation options: XTIOT, UCB NOCAPTURE, and DSAB above the line, except when the calling program supplies an open DCB.

MCDS

ARCUTIL reads the MCDS (you must include a DD statement with a DDNAME of MCDS) to collect data for the following records:

- Migrated-data-set information records
- DASD capacity planning records
- Tape capacity planning records for migration level-2 tapes

BCDS

ARCUTIL reads the BCDS (you must include a DD statement with a DDNAME of BCDS) to collect data for the following records:

- Backup-version information record
- Tape capacity planning record for backup tapes
- Tape capacity planning record for dump tapes

Snap processing data set

To aid in problem determination, the data collection interface allows the application programmer to create a snap dump. The snap processing data set contains this dump. You must include a DD statement with a DDNAME of ARCSNAP to collect data for the snap processing data set.

Collection data set

The collection data set contains the records requested by the application program. One collection data set contains all of the records requested when the data collection interface is invoked. You must include a DD statement with a DDNAME of ARCDATA to refer to this data set. The interface allows the application program to append the DFSMShsm collection records onto the end of an existing collection data set.

The collection data set is a physical sequential data set with a variable or variable-blocked record format. Any record length can be specified provided it is large enough to contain the largest collection record generated. The following is an example of a valid configuration:

```
DSORG = PS
RECFM = VB
LRECL = 264
BLKSIZE = 5280
```

Quick size calculation for the collection data set

If all (and only) DFSMShsm-specific records are requested, use the work sheet shown in Figure 92 to calculate the size (in tracks) for the collection data set.

Collection Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = MCDStracks - Number of tracks used by the MCDS.

   ________ = DCDStracks - Number of tracks used by the BCDS.

2. Substitute the values for *MCDStracks* and *BCDStracks* in the following calculation:

   (MCDStracks = .25 = ________) = ________________

   + (BCDStracks = .35 = ________) = ________________

   Total = **total number of tracks for the collection data set**

ARC1C160

Figure 92. Collection Data Set Size Work Sheet. This worksheet is also found in Appendix A, "DFSMShsm work sheets," on page 387.

Data collection records

The DFSMShsm data collection interface collects data from the following records:

- Migrated-data-set information records
- Backup-version information records
- DASD capacity planning records
- Tape capacity planning records

ARCUTIL then generates a fixed-length header followed by a variable-length area specific to the record type.

Data collection record header

The data collection record is a fixed-length header for all the record types. It contains all the common fields that are needed regardless of the type of data collected. All other output record data is appended to the header.

The data collection record header is available in SYS1.MACLIB(IDCDOUT) provided by DFP Version 3 Release 2.0 (or later) and by DFSMSdfp.

You may access all data collection record definitions (including DFSMSdfp-specific records) in a program written in assembler language as follows: **ARCUTILP IDCDOUT=YES**.

If you wish to define only the header and DFSMShsm-related records, use: **ARCUTILP** or **ARCUTILP IDCDOUT=NO**.

Migrated data set information record

If migrated-data-set information (Type M) records are requested, one record is created for each migrated data set represented in the MCDS.

Backup version information record

If backup-version information (Type B) records are requested, one record is created for each backup version represented in the BCDS.

DASD capacity planning records

If DASD capacity planning (Type C) records are requested, one record is created for each level-0 and level-1 volume for each day there has been activity. For example, if five volumes had DFSMShsm activity for seven days, there would be 35 DASD capacity planning records. The number of days that volume statistics are kept to create these records can be controlled by the MIGRATIONCLEANUPDAYS parameter of the DFSMShsm SETSYS command.

Tape capacity planning records

If tape capacity planning (Type T) records are requested, one record is created for each of the following types of DFSMShsm tapes:

- Migration level-2 tapes
- Incremental-backup tapes
- Full-volume-dump tapes

Both the MCDS and BCDS are needed to create these records. If backup availability is not enabled in the installation, the BCDS DD statement in the job must be specified with a DD DUMMY value.

Related reading

Refer to *z/OS DFSMS Access Method Services Commands* for descriptions of the record header, the migrated-data-set information (Type M) record, the backup-version information (Type B) record, the DASD capacity planning (Type C) record, and the tape capacity planning (Type T) record.

Invoking the DFSMShsm data collection interface

The entry point of the data collection interface is the ARCUTIL load module. The ARCUTIL load module is created when DFSMShsm is installed.

As discussed in "Choosing a data collection method" on page 314, you can invoke data collection in the following three ways:

- Invoking the ARCUTIL load module with the access method services (AMS) DCOLLECT command
- Invoking the ARCUTIL load module directly using JCL
- Invoking the ARCUTIL load module with a user-written program

Invoking the ARCUTIL load module with the access method services (AMS) DCOLLECT function

You can invoke the IDCAMS DCOLLECT function with the access method services DCOLLECT command. For a detailed discussion of the DCOLLECT command, refer to *z/OS DFSMS Access Method Services Commands*.

Figure 93 on page 321 could be used to invoke the DCOLLECT function. The DCOLLECT keywords shown in the example are the keywords that select the record types pertaining to DFSMShsm.

```
//DCOLLECT   JOB   ,'DCOLLECT RUN',CLASS=Z,MSGCLASS=H,REGION=4M
//*
//STEP1   EXEC   PGM=IDCAMS
//*
//SYSPRINT  DD  SYSOUT=*
//ARCSNAP   DD  SYSOUT=*
//MCDS      DD  DSN=HSM.MCDS,DISP=SHR
//BCDS      DD  DSN=HSM.BCDS,DISP=SHR
//DCOUT     DD  DSN=userid.DCOLLECT.OUTPUT,
//              DISP=(NEW,CATLG,DELETE),
//              SPACE=(1,(859,429)),AVGREC=K,
//              DSORG=PS,RECFM=VB,LRECL=264
//SYSIN     DD  *
     DCOLLECT -
           OUTFILE(DCOUT) -
           NODATAINFO -
           NOVOLUMEINFO -
           MIGRATEDATA -
           BACKUPDATA -
           CAPPLANDATA -
           MIGRSNAPERR

/* END OF DCOLLECT COMMAND
```

Figure 93. Invocation of the ARCUTIL Load Module with the Access Method Services DCOLLECT Command. This example JCL program invokes the ARCUTIL load module by using the AMS DCOLLECT command.

Direct invocation of ARCUTIL load module

The ARCUTIL load module can be invoked directly, which can be useful when only DFSMShsm-related data collection records are required. Direct invocation of the ARCUTIL load module is also useful for problem determination.

Options are specified as DCOLLECT keywords on the PARM specified on the EXEC statement. The protocol for the PARM must be the command DCOLLECT followed in any order by one or more of the following keywords:

DCOLLECT Keyword
: **Description**

MIGRATEDATA
: Request migrated data set information **Abbreviations:** MIGRATE, MIGD

BACKUPDATA
: Request backup version information **Abbreviations:** BACKUP, BACD

CAPPLANDATA
: Request DASD and tape-capacity information **Abbreviations:** CAPPLAN, CAPD

MIGRSNAPERR
: Request SNAP if return code is nonzero **Abbreviations:** MSERR

MIGRSNAPALL
: Request SNAP processing **Abbreviations:** MSALL

When you invoke the ARCUTIL load module directly, the following data sets are required:

- ARCTEST

  Specify a DDNAME of ARCTEST that contains the messages from ARCUTIL processing. The following are example messages:

```
DCOLLECT MIGD CAPD BACD MSERR
RETURN CODE....................     0
REASON CODE....................     0
TOTAL RECORDS WRITTEN:
NUMBER OF: MIGRATION DATA...... 45350
NUMBER OF: BACKUP DATA.........316540
NUMBER OF: DASD CAPACITY.......  7090
NUMBER OF: TAPE CAPACITY.......     3
```

- ARCDATA

  Specify a DDNAME of ARCDATA that contains the records collected. ARCUTIL opens this data set with LRECL=264, RECFM=VB options.

Additionally, if you request snap processing by specifying the MIGRSNAPERR and MIGRSNAPALL keywords, ensure that you include a DDNAME of ARCSNAP. Including this DDNAME in your JCL enables DFSMShsm to collect data for the snap-processing data set.

Figure 94 is an example of a job that invokes ARCUTIL and requests all record types and SNAP processing if the return code is nonzero. This is the same as the USRPGM example.

```
//JOB2    JOB accounting information,REGION=nnnnK
//STEP2   EXEC PGM=ARCUTIL,PARM='DCOLLECT MIGD CAPD BACD MSERR'
//ARCSNAP SYSOUT=*
//ARCTEST SYSOUT=*
//ARCDATA DD DSN=MY.COLLECT.DATA,DISP=(,CATLG),
//           SPACE=(CYL,(5,10)),UNIT=SYSDA
//MCDS    DD DSN=HSM.MCDS,DISP=SHR
//BCDS    DD DSN=HSM.BCDS,DISP=SHR
```

Figure 94. Direct Invocation of the ARCUTIL Load Module with JCL. This JCL program example invokes the ARCUTIL load module directly and requests that DFSMShsm-related information be placed in the data collection data set.

Invoking the ARCUTIL load module with a user-written program

You can create custom reports in an environment that does not support MVS/DFP Version 3 Release 2.0 by invoking the ARCUTIL load module with a program written in assembler language.

Example of invoking ARCUTIL with a user-written program: Figure 95 is an example of JCL that invokes the sample application program that is shown in Figure 96 on page 323. The read-only access to the MCDS and BCDS uses a shared disposition, which allows the sample program to run while DFSMShsm is currently active.

```
//JOB1     JOB accounting information,REGION=nnnnK
//STEP1    EXEC PGM=USRPGM
//STEPLIB  DD  DSN=MY.LINKLIB,DISP=SHR
//ARCSNAP  DD  SYSOUT=*
//COLLECT  DD  DSN=MY.COLLECT.DATA,DISP=(,CATLG),
//             SPACE=(CYL,(5,10)),UNIT=SYSDA
//MCDS     DD  DSN=HSM.MCDS,DISP=SHR
//BCDS     DD  DSN=HSM.BCDS,DISP=SHR
//*
```

Figure 95. Invocation of the ARCUTIL Load Module with a User-Written Program. This JCL program example invokes the ARCUTIL load module by using a user-written program.

The sample program in Figure 96 on page 323 opens the collection data set, links to ARCUTIL, and then closes the collection data set. All DFSMShsm-specific collection record types are requested. If the return code is not zero, a snap dump is written to a SYSOUT file.

```
**************************************************************************
*  MODULE NAME:     USRPGM                                                *
*  DESCRIPTION:     DFSMSHSM DATA COLLECTION INTERFACE, PROGRAM EXAMPLE  *
*  FUNCTION:        INVOKE THE DFSMSHSM DATA COLLECTION INTERFACE TO     *
*                   CREATE A SEQUENTIAL FILE THAT CONTAINS INFORMATION   *
*                   ABOUT MIGRATED DATA, BACKED UP DATA, DFSMSHSM DASD   *
*                   VOLUMES AND DFSMSHSM TAPE VOLUMES.                   *
*  DETAILS:                                                              *
*                   THE UPRECORD FIELD IS SET TO REQUEST THE             *
*                   FOLLOWING RECORDS:                                   *
*                        - MIGRATED DATA SET                             *
*                        - BACKUP VERSION                                *
*                        - DASD VOLUME PLANNING                          *
*                        - TAPE VOLUME PLANNING                          *
*                                                                        *
*                   THE UPOPTION FIELD IS SET TO REQUEST A SNAP DUMP     *
*                   IF THE RETURN CODE IS NONZERO.  THE ERROR            *
*                   INFORMATION IS WRITTEN TO A DATASET WITH A DDNAME    *
*                   OF ARCSNAP.                                          *
*                   EXIT SUPPORT IS NOT REQUESTED.                       *
**************************************************************************

USRPGM   CSECT ,
USRPGM   AMODE 24
USRPGM   RMODE 24
*                                     SAVE REGISTERS IN SAVE AREA
         STM   14,12,12(13)
         BALR  12,0
         USING *,12
         LA    3,SAVEAREA
         ST    3,8(13)
         ST    13,4(3)
         LR    13,3
*                                     OPEN COLLECTION DATA SET
         OPEN (DCBOUT,OUTPUT)
*                                     SET PARAMETERS IN ARCUTILP DATA AREA
         LA    2,DCBOUT
         ST    2,UPOUTDCB             SET OUTPUT DCB ADDRESS
         TIME  DEC
         ST    0,UPHTIME              SET TIME AND DATE STAMP
         ST    1,UPHDATE
*
         LA    2,UTILP
         LINK EP=ARCUTIL,PARAM=((2))  INVOKE INTERFACE
*
```

Figure 96. Sample Program for Data Collection Part 1 of 2

```
         CLOSE DCBOUT                 CLOSE COLLECTION DATA SET
         L     15,UPRC                SAVE RETURN CODE
         L     13,4(13)
         L     14,12(13)
         LM    0,12,20(13)
         BR    14                     RETURN TO CALLER
*

*
DCBOUT   DCB DDNAME=COLLECT,LRECL=264,BLKSIZE=5280,DSORG=PS,          X
               MACRF=(PL),RECFM=VB
*
         CNOP  0,4                BEGIN ARCUTILP DATA AREA
UTILP    DC    CL8'ARCUTILP'
UPVERS   DC    X'01'
UPRECORD DC    B'11110000'        REQUEST ALL RECORD TYPES
         DC    XL1'00'
UPOPTION DC    B'01000000'        CREATE SNAP DUMP IF ERROR OCCURS
UPOUTDCB DS    AL4
UPHTIME  DS    CL4
UPHDATE  DS    CL4
         DC    XL20'00'
UPNUMIGR DC    F'0'               NUMBER OF RECORDS WRITTEN
UPNUBACK DC    F'0'
UPNUDASD DC    F'0'
UPNUTAPE DC    F'0'
         DC    XL20'00'
UPRC     DC    F'0'               RETURN CODE
UPREAS   DC    F'0'               REASON CODE
*
SAVEAREA DS    18F                SAVE AREA FOR REGISTERS
*
         END
```

Figure 97. Sample Program for Data Collection Part 2 of 2

Sample REXX program DCOLREXX: DFSMShsm includes a sample REXX program, DCOLREXX, for generating simple data collection reports from the output of ARCUTIL. See Chapter 7, "DFSMShsm sample tools," on page 151 for the way to access this program.

Data collection exit support: Data collection provides the facility to skip, modify, or replace data collection records. The user program can specify the address of an exit and the address of a 100-byte control area. The exit is invoked prior to writing each data collection record. The exit is also supported directly using DCOLLECT, either using IDCDCX1, or the EXIT(exit) keyword.

The data collection exit, shown in Figure 98 on page 325, must be reentrant, and must be written to process below the 16MB line:

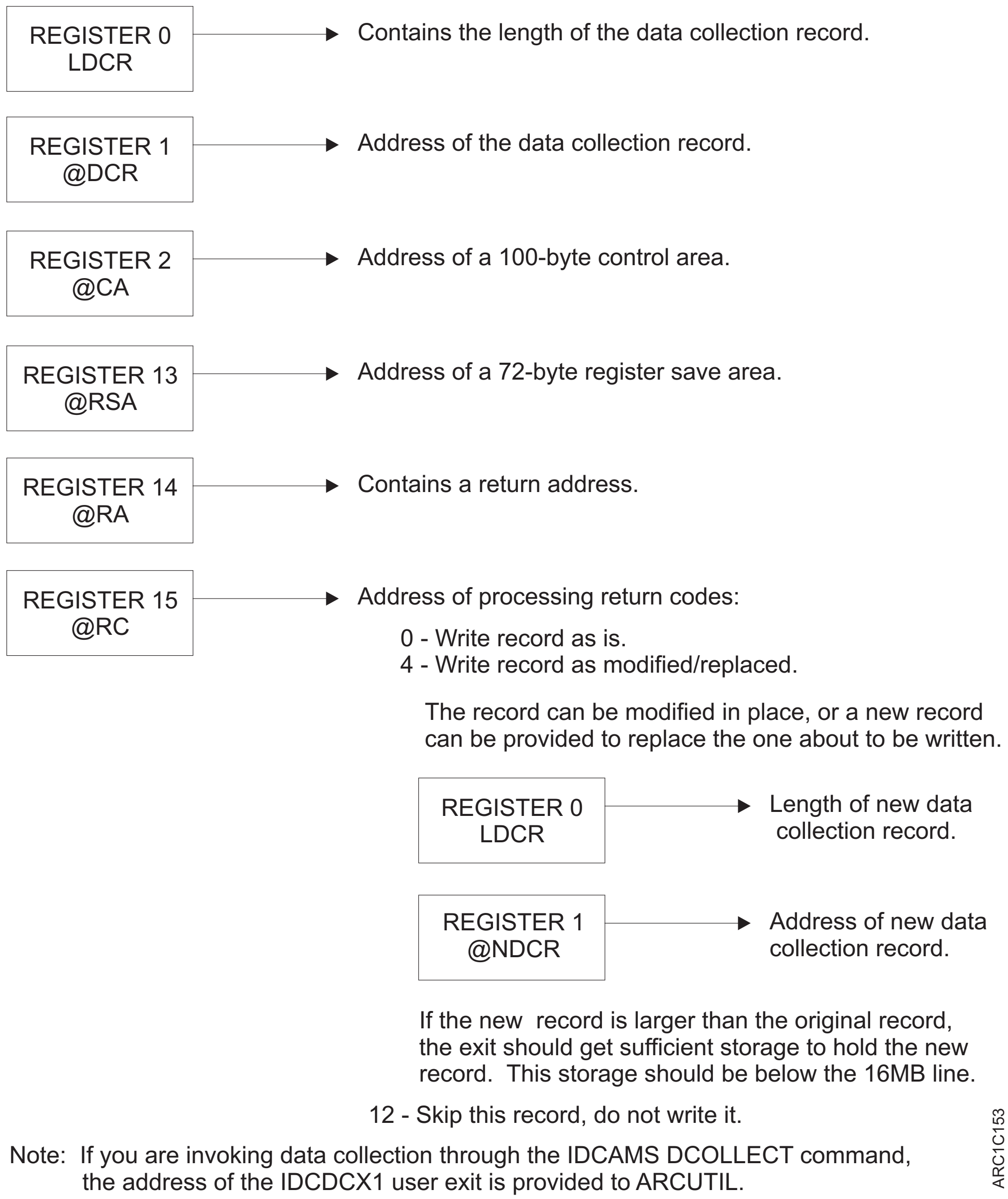


Figure 98. Exit for Data-Collection Support

The following registers are used with the exit described in Figure 98:

- On entry, register 1 contains the address of the parameter list. The parameter list must reside below the 16MB line. The first and only address in the parameter list is an address to the ARCUTILP data area.
- On entry, register 2 contains the address of a 100-byte control area. This 100-byte control area is for the system programmer's use when writing the exit. For example, the work area can contain counters, totals, or other statistics.
- On entry, register 13 contains the address of a save area sufficient to store the program state (72 bytes).
- On entry, register 14 contains the return address.
- On exit, register 15 contains the return code. This return code is also available in the ARCUTILP data area. See Table 49 on page 327 for a description of possible return codes and reason codes.

The ARCUTILP data area: The ARCUTILP data area is a storage area used by DFSMShsm to hold information gathered for the DFSMShsm data collection interface.

The ARCUTILP data area contains input and output parameters to the data collection interface. This data area must reside below the 16MB line.

The ARCUTILP mapping macro for this area is shipped with DFSMShsm and stored as SYS1.MACLIB(ARCUTILP) when DFSMShsm is installed.

Required parameters

Field Name
: **Description**

UPID
: Must be set to the constant shown under UPIDNAME.

UPVERS
: Must contain the version number that identifies this level of ARCUTILP defined by the halfword constant shown under UPVERNUM. If the level of ARCUTIL and ARCUTILP are incompatible, the request fails.

UPRECORD
: Consists of bits for each type of record that can be requested. Set the appropriate bit to 1 (on) to request a particular record type. Any combination of records can be requested, but at least one record type must be requested. See "Data collection records" on page 319 for a description of each record.

UPOUTDCB
: Must contain the address of a data-control block (DCB) for the output data set that contains the collection records. This data set is already open when passed to the data collection interface. The DCB must reside below the 16MB line.

Optional parameters

Field Name
: **Description**

UPSPALL
: Designed for problem determination. Set this bit to 1 (on) to request snap processing. The snap data set contains the registers at entry, followed by the problem program area on completion. This option is useful if the data collection interface returns a zero return code, but not all the records requested are being written to the collection data set.

: UPSALL can also be used to suppress ESTAE protection against abnormal ends. If you receive an RC=20, specify UPSALL='1'b and a DD card for SYSUDUMP. The SYSUDUMP can be dumped to SYSOUT or a DASD data set and contains a formatted dump of the abnormal end.

: UPSPALL and UPSPERR are mutually exclusive parameters; do not specify both parameters.

UPSPERR
: Designed for problem determination. Set this bit to 1 (on) to request snap processing only when the return code is nonzero. UPSPERR and UPSPALL are mutually exclusive parameters; do not specify both parameters.

UPSTAMP
This field contains a time and date stamp that is copied into the data collection header for each record generated.

UPEXITP
If exit support is desired, set this field to the address of an exit to be called before writing each data collection record.

UPEXAREA
If exit support is desired, set this field to the address of a 100-byte control area that is passed to the exit. This area can be used for control parameters, counters, or anchors to larger data structures.

Output parameters

Field Name
Description

UPNUMIGR
Contains the number of migrated-data-set information records written by the data collection interface.

UPNUBACK
Contains the number of backup-version information records written by the data collection interface.

UPNUDASD
Contains the number of DASD capacity planning records written by the data collection interface.

UPNUTAPE
Contains the number of tape capacity planning records written by the data collection interface.

UPRC Return code.

UPREAS
Reason code.

Return codes and reason codes are described in Table 49.

ARCUTIL return codes and reason codes

Table 49 is a summary of the return codes and reason codes issued by the ARCUTIL load module.

Table 49. Return Codes and Reason Codes

Return Code	Reason Code	Description
0	--	Function successfully completed.
4	--	Invalid parameter list.
	1	UPID not equal to UPIDNAME.
	2	UPVERS incompatible with current version of ARCUTIL.
	3	DCB address not provided.
	4	No function requested to perform.
	5	Invalid combination of options.
8	--	Error opening DFSMShsm control data set.
	1	DFSMShsm MCDS cannot be opened
	2	DFSMShsm BCDS cannot be opened

Table 49. Return Codes and Reason Codes (continued)

Return Code	Reason Code	Description
12	--	Error reading DFSMShsm control data set.
	1	Over 1% of DFSMShsm MCDS records required cannot be read.
	2	Over 1% of DFSMShsm BCDS records required cannot be read.
	3	User specified DUMMY on the MCDS or BCDS DD card. Each DD statement must contain the data set name of the corresponding DFSMShsm migration control data set or backup control data set. The specification of DUMMY in this field is not accepted.
	11	Position error to MCDS record occurred
	12	Position error to BCDS record occurred.
16	--	Error writing to output data set. The reason code contains the contents of register 1 received from the SYNADAF macro.
20	--	Abnormal termination occurred. For additional information on problem determination, see UPSPALL under "Optional parameters" on page 326.
	X'ccSSSuuu'	The reason code contains the abnormal termination code. *SSS* is system abend code *uuu* is user abend code Example: 040C4000 is a system 0C4 abend.
24	--	Internal processing error
	1	SNAP processing is requested, but the data set specified by the ARCSNAP DD cannot be opened successfully.

Chapter 16. Tuning DFSMShsm

This topic is intended to help the customer to tune DFSMShsm with supported tuning patches. This topic documents diagnosis, modification, or tuning information, which is provided to help the customer to tune DFSMShsm. Not all tuning information is contained in this topic—only that related to the DFSMShsm supported patches.

For sites with unique requirements of DFSMShsm that are not supported by existing DFSMShsm commands and parameters, these tuning patches may offer a solution. Where those unique requirements remain the same from day to day, the installation may choose to include the patches in the DFSMShsm startup member. These DFSMShsm-supported patches remain supported from release to release without modification.

The supported patches are described in "Tuning patches supported by DFSMShsm." For more information on using the PATCH and DISPLAY commands, see *z/OS DFSMShsm Diagnosis*.

Guidelines

Before applying any patches to your system, be aware of the following:

- Some of the PATCH commands given in this section include the VERIFY parameter or comments about the patch. The VERIFY parameter and comments are optional. However, when you are patching full bytes of data from a terminal, use the VERIFY parameter to help catch any errors in the command entry. To see the current value of a byte before changing it, use the DISPLAY command.
- If you are using the PATCH command to change only part of a byte, use the BITS parameter.
- If you need to see the output data from a PATCH command online, you can specify the OUTDATASET parameter of the PATCH command before you shut down DFSMShsm.
- If you are running multiple instances of DFSMShsm in a single z/OS image, you may need to repeat the PATCH command for each DFSMShsm host. Repeat the PATCH command if you are patching any of the following records: MCVT, BGCB, MGCB, YBCB, BCR, DCR, MCR. If the PATCH commands are in the ARCCMD*xx* PARMLIB member, use the ONLYIF HSMHOST(*x*) command to restrict specific hosts.

Tuning patches supported by DFSMShsm

This topic describes tuning patches that are supported by DFSMShsm.

Changing DFSMShsm backup and migration generated data set names to reduce contention for similar names and eliminating a possible performance degradation

In environments where many small similarly named data sets are backed up or migrated across multiple tasks and multiple instances of DFSMShsm, a possible contention for the target data set name may occur. This contention could cause a performance degradation. DFSMShsm normally creates unique data set names for its backup and migrate data sets using the form:


```
prefix.function.Tssmmhh.user1.user2.Xyddd
```

where *prefix* is the backup or migration defined prefix, *function* is either BACK or HMIG, T*ssmmhh* is a timestamp, *user1* and *user2* are the first two high-level qualifiers of the source dataset name, and X*yddd* is a date.

After the feature is enabled, in generated names T*ssmmhh* is replaced with T*cccchh*, where *cccc* is the time in hundredths of seconds from the beginning of the hour, converted to four alphabetic characters, and *hh* is the hour.

See *z/OS DFSMShsm Storage Administration*. for more details and conversion information.

To activate the feature, issue the following:

```
PATCH .MCVT.+24C BITS(.......1)
```

To deactivate the feature, issue the following:

```
PATCH .MCVT.+24C BITS(.......0)
```

Migrating and scratching generation data sets

Catalog routines uncatalog non-SMS generation data sets at roll-off time. Nonmigrated generations are scratched by DFSMSdfp. Migrated generations are scratched by DFSMShsm. The process of deleting migrated, rolled-off generation data sets depends on the SVC number used to access the IBM catalog routines. SVC 26 (X'1A') must be used to call the IBM catalog routines. The catalog routines call the scratch function to delete the rolled-off generation data sets. If the SVC number used to invoke the IBM catalog routines is changed from SVC 26, DFSMShsm does not recognize the roll-off situation, and the migrated, rolled-off generation data sets are not deleted.

Migrating generation data sets

DFSMShsm does not support migration of non-SMS password-protected generation data sets. There are environments, however, where an installation may need to migrate password-protected GDSs. DFSMShsm, therefore, provides a PATCH command to allow password-protected, non-SMS generation data sets to be migrated, but then scratches them at roll-off time without checking the password.

Allowing password-protected generation data sets to be migrated, then scratched without checking the password at roll-off time: When the MCVTFPW bit is set to 1, DFSMShsm allows password-protected generation data sets to be migrated. Then, when generations are rolled off, the generation data sets are scratched without the password being checked. To set this bit to 1, enter the following command:

```
PATCH .MCVT.+53 BITS(.1......) /* allow pswd protected GDS to migrate then  */
                               /* scratch without checking pswd at roll-off */
```

This allows DFSMShsm to migrate password-protected generation data sets, and also allows DFSMShsm to ignore the password requirement when it scratches the oldest generation of a password-protected generation data set that has been migrated at the time of roll off.

Scratching of rolled-off generation data sets

Scratching of rolled-off generation data sets is performed by DFSMSdfp and DFSMShsm. The generation data set SCRATCH or NOSCRATCH option and the existence of an expiration date control whether DFSMSdfp and DFSMShsm can scratch rolled-off members:

- *If you want to scratch rolled-off members of GDGs* with DFSMSdfp and DFSMShsm, define your GDGs with the SCRATCH option.
  - DFP scratches nonmigrated generations when they roll off.
  - DFSMShsm scratches migrated generations when they roll off if no expiration date exists for the data set.
  - DFSMShsm scratches migrated generations as they roll off if they have an expiration date and if that expiration date has passed. If migrated generations roll off and their expiration date has not passed, the generations are deleted as a part of migration cleanup after the expiration date has passed. For performance reasons, the expiration date should come after the date that you expect the generation to roll off. If the data set has expired but not rolled off, extra catalog processing is required each time migration cleanup runs until that data set rolls off.
- *If you do not want to scratch rolled-off members of GDGs* at all, define the GDGs with the NOSCRATCH option and do not specify expiration dates for the generations.
  - DFP does not scratch nonmigrated rolled-off generations because of the NOSCRATCH option.
  - DFSMShsm does not scratch the migrated rolled-off generations because it is not notified that they are to be scratched and because they never expire.
- If you want to scratch rolled-off members of GDGs when they are defined with the NOSCRATCH option, you need to specify expiration dates for the generations.

  DFSMShsm scratches migrated rolled-off generations during migration cleanup if the data set has an expired expiration date, even if you specify the NOSCRATCH option. For performance reasons, the expiration date should come after the date that you expect the generation to roll off. If the data set has expired but has not rolled off, extra catalog processing is required each time migration cleanup runs until the data set rolls off.

Scratching non-SMS generation data sets at roll-off time regardless of expiration dates: The MCVTFGDG patch only applies to non-SMS generation data sets. The management class controls SMS generation data sets. For more information about specifying GDG management attributes and deleting expired data sets, see the chapter on space management of SMS-managed storage in *z/OS DFSMShsm Storage Administration*.

When the MCVTFGDG bit is set to 1, DFSMShsm ignores the expiration date of a migrated generation data set at roll-off time, and immediately scratches the data set. To set this bit to 1, enter the following command:

```
PATCH .MCVT.+53 BITS(1.......) /* ignore expiration date at GDS roll-off    */
                               /* and scratch the data set.                 */
```

When this command is issued, migrated, date-protected, and rolled-off generation data sets defined with the SCRATCH option are scratched as the generation rolls off.

To allow both the migration of password-protected GDSs with their deletion at roll off without password checking, and the deletion of unexpired, date-protected GDSs at roll-off time, issue the following command:

```
PATCH .MCVT.+53 BITS(11......) /* allow pswd protected GDS to migrate then  */
                               /* scratch without checking pswd at roll off */
                               /*                   and                     */
                               /* ignore expiration date at GDS roll off    */
                               /* and scratch the data set.                 */
```

Generation data set scratch summary: DFSMShsm automatically scratches rolled-off generations at roll-off time when:

- A non-date-protected generation data set has the SCRATCH option.
- DFSMShsm is patched for immediate scratch of migrated, date-protected, generation data sets with the SCRATCH option.

DFSMShsm automatically scratches rolled-off generations during migration cleanup when:

- DFSMShsm is not patched for immediate scratch at roll off time of migrated, date-protected, generation data sets defined with the SCRATCH option and the expiration date is later met during migration cleanup.
- A date-protected generation data set has the NOSCRATCH option and the expiration date is met.

Guideline: If you need to use the preceding PATCH commands, be aware of the additional responsibility associated with their use. DFSMShsm is designed to maintain the security of data that is provided by expiration dates and passwords. If you use these commands, you are using DFSMShsm in a way that is not considered to be part of its normal implementation.

Disabling backup and migration of data sets that are larger than 64K tracks to ML1 volumes

You can disable DFSMShsm backup and migration of data sets that are larger than 64K tracks to ML1 volumes by issuing the following patch:

```
PATCH .MCVT.+595 BITS(.......1)
```

By default, data sets larger than 64K tracks are backed up and migrated to ML1 volumes.

Disabling, in JES3, the delay in issuing PARTREL (partial release) for generation data sets

In a JES3 sysplex, DFSMShsm normally delays across two midnights before issuing a PARTREL (directed by management class) to release over-allocated space in new generation data sets. If a JES3 account protects its data sets by using multiprocessor serialization (sending the SYSDSN resource around the GRS ring) rather than using JES3 data set reservation, this delay is unnecessary. To disable this delay, enter the following command:

```
PATCH .MGCB.+27 BITS(.......1)
```

Using DFSMShsm in a JES3 environment that performs main device scheduling only for tapes

The MCVTJ25T bit can be used in JES3 environments that do main device scheduling only for tapes. When this patch is applied, your environment is considered to be a JES2.5 system. The benefit of using the MCVTJ25T bit is that data can be shared between a JES2 and a JES3 system. The following functions result from applying this patch:

- A volume serial number of MIGRAT and a DASD device type are returned for migrated data sets. There are no directed recall setups data sets migrated to tape.
- There are no prevent-migrates during JES3 setup.
- The length of console messages is controlled by the JES3 maximum message length (independent of the JES specification with the SETSYS JES2 or SETSYS JES3 command).

Users applying this patch may want to tell DFSMShsm that the environment is JES2 not JES3. This is not required for the previously listed functions, but it is required to remove restrictions on volume configuration changes (for example ADDVOLs of level-0 volumes restricted to startups associated with DFSMShsm JES3).

This patch can be added to your DFSMShsm startup member. If you want only to test the patch, it can be entered from a terminal. If you enter the patch from a terminal, some command (for example RELEASE ALL) must be received to cause the patched MCVT value to be propagated to CSA (MQCT) where SVC 26 (IGG026DU) can see it. Enter the PATCH command as follows:

```
PATCH .MCVT.+29A BITS(1.......) /* run as JES2.5 for JES3 using main */
                                /* device scheduler only for tape.   */
```

Note: You cannot use this patch when doing main device scheduling for DASD unless you have a mounted DASD volume with a volume serial number of MIGRAT.

Shortening the prevent-migration activity for JES3 setups

DFSMShsm delays migration for any non-SMS-managed data sets that have been processed by the JES3 setup function. By delaying their migration, DFSMShsm ensures that non-SMS-managed data sets are not migrated between the time that they are processed by JES3 setup and the time that the job is actually run.

DFSMShsm controls migration prevention of non-SMS-managed data sets that have been processed by the JES3 setup function by accepting the defaults for the MCVTJ3D byte. The defaults also apply to the period of time that a recalled data set is prevented from migrating.

The DFSMShsm default for this byte is X'03', which means that migration is prevented for the remainder of the processing day plus three more days. Through the delayed migration DFSMShsm ensures the data integrity of non-SMS-managed data sets in a JES3 environment. Without the migration delay, the small possibility exists of a data integrity problem.

There can be a data integrity problem only if the following sequence of events has occurred:

1. On a system not using a global resource serialization (GRS) function for the SYSDSN resource, job 1 goes through JES3 setup referring to a particular nonmigrated data set.
2. The subject data set is migrated before job 1 runs and before job 2 goes through JES3 setup.

   The minimum age to be eligible for migration in this nonglobal resource serialization environment is one complete day plus the remainder of the day on which the JES3 setup has occurred. Most jobs run before the data set has aged enough to be selected by automatic volume migration. Instances of the subject data set migrating before job 1 runs are most likely caused by command migration.
3. Job 2 goes through JES3 setup and also refers to the subject data set, but now the subject data set is in the migrated state. As a result, DFSMShsm returns a list of volumes to JES3 that is different from the list provided for job 1.
4. JES3 performs its data set reservation function but sees this single data set as two different data sets, because the combinations of data set name and volume serial numbers are different.
5. Jobs 1 and 2 run on different CPUs concurrently. An update is lost when the subject data set is accessed in the following sequence:
   a. Read for update from one CPU.
   b. Read for update from the other CPU.
   c. Write the first update from one CPU.
   d. Write the second update (over the top of the first update) from the other CPU.

   The preceding is known as a read, read, write, write sequence.

There can be times that you do not want DFSMShsm to prevent migration. For example, users who are severely constrained on DASD space can benefit from preventing or shortening the duration of the migration-prevention activity. Without the migration-prevention activity, users who know that particular groups of data sets do not need to be accessed in the immediate future are able to command migrate those groups of data sets.

Another possible benefit in not having migration-prevention activity could be in performance. A significant amount of I/O to the MCDS is related to migration prevention activity; without migration prevention, the system would be free of those I/O tasks.

In the following example, the MCVTJ3D byte is set to X'00'. When the MCVTJ3D byte is set to X'00', DFSMShsm does no prevent migration processing.

```
PATCH .MCVT.+14A X'00' VERIFY(.MCVT.+14A X'03') /*no JES3 migration prevention*/
```

In the following example, the MCVTJ3D byte is set to X'01' to *shorten* migration prevention to the processing day plus one day:

```
PATCH .MCVT.+14A X'01' VERIFY(.MCVT.+14A X'03') /*delay migration for 1 day*/
```

Replacing HSMACT as the high-level qualifier for activity logs

The MCVTACTN bytes can be modified to change the high-level qualifier of the DASD activity logs created by DFSMShsm. DFSMShsm supplies a high-level qualifier of HSMACT. Users who have a controlled data set naming convention that is not compatible with this qualifier can change the 7-byte MCVTACTN to create a high-level qualifier of their own choosing.

The MCVTACTN can contain any valid data set name high-level qualifier comprising from 1 to 7 characters. The high-level qualifier should be left justified. If shorter than 7 bytes, the unwanted bytes should be blanked out.

The following example causes DFSMShsm to create its DASD activity logs with the high-level qualifier of SHORT:

```
PATCH .MCVT.+321 'SHORT ' VERIFY(.MCVT.+321 'HSMACT')
```

Note: The 6-character qualifier HSMACT is being replaced by a 5-character qualifier SHORT, so SHORT is followed by a blank to erase the letter T from HSMACT.

Changing the allocation parameters for an output data set

Several DFSMShsm commands allow an optional parameter for directing output to an OUTDATASET (ODS) as an alternative to writing to SYSOUT. You can modify how such an OUTDATASET is allocated, by changing any or all of the following parameters.

Changing the unit name

You can specify any valid unit name, using an eight-character patch. (If the name is less than 8 characters, include one or more trailing blanks.) An example:

```
PATCH .MCVT.+200 '3390    ' VERIFY(.MCVT+200 'SYSALLDA')
                             /* Unit name for ODS allocation */
```

DFSMShsm uses a default unit name of SYSALLDA.

Note:

1. If you use the OUTDATASET option for this PATCH command, the patch does not affect that allocation.
2. The PATCH command does not verify that the unit name is valid on your system.

Changing the primary space quantity

You can specify the number of tracks to be allocated as the primary space quantity for the specified (or default) unit. An example:

```
PATCH .MCVT.+4B0 X'001E' VERIFY(.MCVT.+4B0 X'0014')
/* Primary space qty for ODS allocation = 30 tracks */
```

DFSMShsm uses a default primary allocation of 20 tracks.

Note: If you use the OUTDATASET option for this PATCH command, the patch does not affect that allocation.

Changing the secondary space quantity

You can specify the number of tracks to be allocated as the secondary space quantity for the specified (or default) unit. An example:

```
PATCH .MCVT.+4B2 X'0028' VERIFY(.MCVT.+4B2 X'0032') /* Secondary space      */
                                                    /* quantity for ODS     */
                                                    /* allocation=40 tracks */
```

DFSMShsm uses a default secondary allocation of 50 tracks.

Note: If you use the OUTDATASET option for this PATCH command, the patch does not affect that allocation.

Changing the limiting of SYSOUT lines

If you specify a SYSOUT class instead of OUTDATASET, you can change whether DFSMShsm limits the lines of SYSOUT or specify what that limit is by using the following patch:

```
PATCH .MCVT.+1BD X'3D0900'  /* Limit SYSOUT size to 4 million lines */
```

The maximum allowable value is 16777215 (X'FFFFFF'). If you specify X'000000', DFSMShsm does not specify limiting for SYSOUT; the result is determined by how your JES2 or JES3 system is set up. For more information, see *z/OS JES2 Initialization and Tuning Reference* or *z/OS JES3 Initialization and Tuning Reference*.

If you specify a nonzero value, it must be greater than or equal to the DFSMShsm default limit of 2000000 (X'1E8480') lines; otherwise, DFSMShsm uses that default limit.

Note: If you use a limiting value (including the default), but a SYSOUT data set exceeds that limit, the system cancels DFSMShsm.

Using the DFSMShsm startup PARMLIB member

If you want to take advantage of these patches, you should include the relevant PATCH command or commands in the startup member. If these patches are not entered, DFSMShsm uses its default values to allocate each OUTDATASET and to limit SYSOUT files.

Buffering of user data sets on DFSMShsm-owned DASD using optimum DASD blocking

DFSMShsm defaults to optimum DASD blocking (18KB blocks for 3390 devices) when accessing user data sets in DFSMShsm format on its owned DASD. Buffers are obtained above the 16MB line.

In addition, two fields in the MCVT provide the default number of buffers DFSMShsm uses to write (field MCVTOBUF, offset X'393') and read (field MCVTNBUF, offset X'392') such user data. If you have previously patched the MCVT offset X'390' for this purpose, you should remove this patch.

Changing parameters passed to DFSMSdss

DFSMShsm by default uses certain parameters when invoking DFSMSdss for certain functions. You can use the patches below to change how those parameters are used.

Invoking DFSMSdss for a full-volume dump

When DFSMShsm invokes DFSMSdss to perform a full-volume dump, DFSMShsm causes the dump to be done specifying the ALLEXCP and ALLDATA parameters but *not* specifying the COMPRESS parameter. Patches can be applied either to add or to remove any of these three functions.

Setting on compress for dumps and ABARS: A patch can be applied to specify that volume dumps are to be performed with the COMPRESS parameter. The advantage of compressing data is that the dump usually requires fewer output tapes. There are, however, two points to consider before applying the patch: 1) performing the compress function during a volume dump requires more CPU time and probably more elapsed time, and 2) dump volumes created with the COMPRESS parameter cannot be restored with some levels of the stand-alone version of DFSMSdss.

Among the types of volumes that may need to be restored with a stand-alone version of DFSMSdss are IPL volumes. To allow users the ability to dump data volumes using the COMPRESS parameter and to dump IPL volumes without using the COMPRESS parameter, DFSMShsm dump classes can be selected to suppress the COMPRESS parameter. Dumps made from the selected dump classes will not be compressed.

To specify that volume dumps be performed with the COMPRESS function, enter the PATCH command as follows:

```
PATCH .MCVT.+3C3 BITS(..1.....) /* use DFSMSdss COMPRESS for dumps */
```

Note:

1. If DFSMShsm is running with a release of DFSMSdss where COMPRESS and OPTIMIZE are mutually exclusive parameters of the DFSMSdss DUMP command, and if a dump is made using the COMPRESS parameter, DFSMShsm overrides the OPTIMIZE value specified by the SETSYS DUMPIO parameter.
2. The patch to use DFSMSdss COMPRESS for dumps also applies to ABARS when performing logical data set dumps during aggregate backup. If ABARS is running with a release of DFSMSdss where COMPRESS and OPTIMIZE are mutually exclusive parameters, the dump is made using the COMPRESS option. Otherwise, the dump is invoked using both OPTIMIZE and COMPRESS.
3. If the HWCOMPRESS keyword is specified for this dump through its dump class, the patch to use COMPRESS is ignored.
4. If the ZCOMPRESS keyword is specified for this dump through its dump class, and the patch to use COMPRESS is specified, then DFSMShsm specifies both the ZCOMPRESS(PREFERRED) and the COMPRESS keywords in the DFSMSdss DUMP command. If zEDC hardware is available, the dump is invoked using the ZCOMPRESS function. In case of a zEDC hardware failure, the dump is invoked using the COMPRESS function.

To specify that COMPRESS not be used for a certain dump class requires a patch to the dump class record at offset X'01'. The patch is required only if the DFSMShsm default has been changed to invoke COMPRESS during volume dumps. This CDS record change remains in effect until changed by a subsequent FIXCDS PATCH command, and it does not have to become part of a startup member. For more information on the FIXCDS command, see *z/OS DFSMShsm Diagnosis*.

In the following example the dump class IPLVOLS is being patched to suppress the COMPRESS parameter:

```
FIXCDS W IPLVOLS PATCH(X'01' BITS(1.......)) /* dump without compress.*/
```

Note: If a volume is dumped to multiple dump classes concurrently and any of the dump classes specify that the COMPRESS parameter is not to be used, then COMPRESS will not be used. No dumps from that dump generation will be compressed.

To determine whether a particular dump generation was made with or without COMPRESS, use FIXCDS DISPLAY to display its dump generation (DGN or G) record. If the X'04' bit in the 1-byte field at displacement X'00' in the variable portion of the record is on (the bit is set to 1), the dump was made with COMPRESS. The following example requests that the G record be displayed for the dump generation made from a source volume with serial number ESARES at 15.78 seconds past 1:44 p.m. on day 325 of year 1989:

```
FIXCDS G X'C5E2C1D9C5E2134415780089325F'
```

If the exact time the dump generation was made is not known, you may want to use the AMS PRINT command to print all the G records for dump generations of the volume in question directly from the BCDS. If the BCDS data set (cluster) name is DFHSM.BCDS, the following commands can be used under TSO to display the G records for the volume with serial number ESARES:

```
ALLOC DA('DFHSM.BCDS') FILE(BCDS) SHR
PRINT IFILE(BCDS) FKEY(X'29C5E2C1D9C5E2') TOKEY(X'29C5E2C1D9C5E2')
```

If this is done, the displacement to the field to be examined is X'40', not X'00'

Setting off ALLEXCP for dumps: To *remove* the ALLEXCP function from the volume-dump function, enter the following PATCH command:

```
PATCH .MCVT.+3C3 BITS(1.......) /* don't use DFSMSdss ALLEXCP */
```

Setting off ALLDATA for dumps: To *remove* the ALLDATA function from the volume-dump function, enter the following PATCH command:

```
PATCH .MCVT.+3C3 BITS(.1......) /* don't use DFSMSdss ALLDATA */
```

Requesting DFSMShsm volume dumps with a 32 KB block size: DFSMShsm provides a patch to the MCVT so that when volume dump is run, the requested block size for DFSMSdss is 32 KB rather than the DFSMSdss system-derived block size.

The 32 760 block-size value is also included in the tape label. This can be used if a following application such as a file transfer required this format. You can use this patch with DFSMSdss.

Note: DFSMSdss only supports block size of 65 520 and above. The default block size for output records that are written to tape is the optimum block size for the output device (262 144 is the maximum).

There is no patch available for DFSMShsm volume dumps with 64 KB block size. If you require this size, add TAPEBLKSZLIM=65520 to the DEVSUP*xx* PARMLIB member. Note that this change will affect all tape volumes for which the BLKSIZE parameter in the tape DD statement is not specified.

The PATCH command is as follows:

```
PATCH .MCVT.+3D4 BITS(.......1) /* change the DFSMSdss blocksize */
                                /* default to 32K.               */
```

Processing partitioned data sets with AX cells

To *use* the ALLDATA function for migrating or recalling a partitioned data set with AX cells, enter the following PATCH command:

```
PATCH .MCVT.+3C3 BITS(....1...) /* Use DFSMSdss ALLDATA when     */
                                /* migrating PO DS with AX cells */
```

Enabling ABARS ABACKUP and ARECOVER to wait for a tape unit allocation

When ABARS processing backs up or recovers an aggregate, a tape unit is requested by ABARS and allocated by MVS. If a request for a tape unit cannot be satisfied (because all tape units are presently in use), the ABACKUP or ARECOVER processing fails.

For all other DFSMShsm processing that uses tape units, a WAITor NOWAIT option can be specified with the SETSYS INPUTTAPEALLOCATION and SETSYS OUTPUTTAPEALLOCATION commands. ABARS processing, however, runs in its own address space and is not effected by the SETSYS commands.

To enable the operator to specify whether to wait for an available tape unit, you must issue a PATCH command.

Issue the following PATCH command if you want the operator to specify whether to wait for tape unit allocation for ABACKUP processing:

```
PATCH .ABRCB.+81 BITS(...1....)        /* Allow the operator to specify */
                                       /* the WAIT option for ABACKUP   */
                                       /* processing.                   */
```

Issue the following PATCH command if you want the operator to specify whether to wait for tape unit allocation for ARECOVER processing:

```
PATCH .ABRCB.+81 BITS(....1...)        /* Allow the operator to specify */
                                       /* the WAIT option for ARECOVER  */
                                       /* processing.                   */
```

Changing the RACF FACILITY CLASS ID for the console operator's terminal

Because not all of your DFSMShsm-authorized storage administrators are generally involved with disaster recovery, you should restrict the ability to issue the ABACKUP and ARECOVER commands. You can restrict who can issue ABACKUP and ARECOVER commands when you specify a RACF FACILITY CLASS and that FACILITY CLASS is active. When you create the FACILITY CLASS, you must explicitly authorize users who have authority to issue these commands; terminal operators (as a group) are authorized when you specify the OPER ID on the RACF PERMIT command, but you can authorize them with a different ID (that you choose) if you use the following PATCH command. For more information about the OPER ID for RACF FACILITY CLASSES, see *z/OS DFSMShsm Storage Administration*. For more information about RACF FACILITY CLASSES, see "Creating the RACF FACILITY class profiles for ABARS" on page 177.

Some sites have used the OPER ID for other operations and those sites need a way to define the operator's terminal to the RACF FACILITY CLASS. By issuing the following PATCH command, you can direct RACF to recognize an ID other than OPER as the authorized ID for the console operator's terminal.

```
PATCH .ABRCB.+4BC 'userid'              /* Change the RACF FACILITY     */
                                        /* CLASS ID that defines the    */
                                        /* console operator's terminal. */
```

Note: *userid* is defined as eight characters. The first seven characters contain the user ID to be used and the eighth character is a blank.

Handling independent-software-vendor data in the data set VTOC entry

The 4-byte area beginning at offset X'4E' of the data set VTOC entry is defined as reserved with DFP releases prior to DFP Version 3. Starting with DFP Version 3, this field has been assigned meanings. Some customer accounts, however, have placed data into the field at offset X'4E', sometimes through ISV software. If altered data set VTOC entries are allowed to be interpreted by DFP Version 3 definitions, unpredictable results can occur.

Before installing DFP Version 3 on systems with altered data set VTOC entries, customers usually prevent problems on the user (level-0) volumes by clearing the field at offset X'4E'. Migrated and backed up data sets, however, must be handled differently. The following paragraphs tell how the DFSMShsm functions of migration and recall can be manipulated to handle the altered data set VTOC entries. The information can also be applied to the DFSMShsm functions of incremental backup and recovery, which are processed in a similar fashion.

When running in a *DFP Version 2* environment, the unmodified code assumes that the data set VTOC entries have not been altered. If the fields at offset X'4E' have been altered but later cleared on the user volumes, the user should supply DFSMShsm with the cutover date when the fields were cleared. Then, when DFSMShsm recalls data sets, those data sets migrated before the cutover date will have the field at offset X'4E' automatically cleared by DFSMShsm. In cases where the customer has a temporary need to preserve the data at offset X'4E', a PATCH command can be used to turn on a bit that allows customers to recall the data set without the field being cleared. If this patch is used, it should be used in addition to setting the previously mentioned cutover date.

When running in a *DFP Version 3* environment, the option to retain the altered field at offset X'4E' of the data set VTOC entry does not exist, but the option to specify a cutover date does. If the fields at offset X'4E' have been altered but later cleared on the user volumes, the user should supply DFSMShsm with the cutover date when the fields were cleared. DFSMShsm then ignores the values at offset X'4E' in the data sets migrated before the cutover date and will clear that field during DFSMShsm recalls of those same data sets. DFSMShsm uses the values in the field at offset X'4E' during the recall of data sets migrated on or after the cutover date.

The field that contains the cutover date is in the MCVT control block at offset X'444'. This field is shipped containing the date of January 1, 1970. In the following example, the PATCH command is used to set the cutover date to July 4, 1989:

```
PATCH .MCVT.+444 X'0089185F' VERIFY(.MCVT.+444 X'0070001F')
```

If you are running with DFP Version 2 on January 1, 1990 and you need to have the values in the altered DSCBs returned during recalls, then you need to set on the bit at offset X'431' of the MCVT. You also need to set the cutover date at offset X'444' of the MCVT. In the following example, two patches are applied. The first patch allows the data in the altered data set entries to be returned to the user during recalls. The second patch sets a cutover date of May 1, 1990, a date when the user expects to have all the altered data cleaned out of the data set VTOC entries.

```
PATCH .MCVT.+431 BITS(.....1..)


PATCH .MCVT.+444 X'0090121F' VERIFY(.MCVT.+444 X'0070001F')
```

Allowing DFSMShsm automatic functions to process volumes other than once per day

To understand the topic of running automatic functions multiple times a day, you must understand the way in which DFSMShsm defines a “day” to these functions. Because automatic processing can begin, for example, at 10 p.m. on one calendar day, and end at 2 a.m. the next calendar day, calendar days cannot be used to define a “day”. Therefore, a volume is defined as having been processed by primary space management, automatic backup, or automatic dump that “day” if any processor has completed processing the volume for the requested function within the past 14 hours.

Running automatic primary space management multiple times a day in a test environment

If automatic primary space management has run to completion and you want to start automatic primary space management again, use the following procedure:

1. Issue one or both of the following PATCH commands:
   - For SMS volumes:

     ```
     PATCH  .MCVT.+414 X'00000000'       /* Specify a new minimum processing */
                                         /* interval for SMS volumes         */
                                         /* processed by automatic           */
                                         /* primary space management.        */
     ```

   - For non-SMS volumes:

     ```
     PATCH  .MCVT.+488 X'00000000'       /* Specify a new minimum processing */
                                         /* interval for non-SMS             */
                                         /* volumes processed by automatic   */
                                         /* primary space management.         */
     ```

2. Issue the SETSYS PRIMARYSPMGMTSTART command to define a new start window. Specify a process window that has a planned start time after the time

that automatic primary space management last ended. You can determine the time that automatic primary space management last completed by examining the migration activity log.

Automatic primary space management starts immediately if the current time is in the new process window. If the current time is not in the new process window, automatic primary space management starts automatically when the planned start time occurs.

Example of redefining a day for automatic primary space management of SMS volumes: There are times when a user may want to modify the definition of a processing day to enable automatic processing to occur more often or less often. By changing the bytes that define the minimum amount of time between automatic processing, you can permit automatic functions to be performed at a different frequency. For example, a user may want a volume to be processed more frequently than every 14 hours. If you were causing DFSMShsm automatic primary space management to occur at the next hour, for example 10 a.m. and you wanted to process all the volumes that had not been processed in the past 30 minutes, perform the following two steps:

1. Apply the following patch to allow automatic primary space management to process the SMS volumes that have not been processed within the last 30 minutes:

```
PATCH .MCVT.+414 X'00000708' VER(.MCVT.+414 X'0000C4E0')
```

2. Set the PRIMARYSPMGMTSTART time to request automatic primary space management to begin at 10 a.m. The following is an example of a PRIMARYSPMGMTSTART command with a planned start time of 10:00 and a stop time of 10:45:

```
SETSYS PRIMARYSPMGMTSTART (1000 1045)
```

In another instance a user may want to migrate all the volumes of a large storage group, but not have enough time available in one day to process all the volumes. By processing the volumes less often, such as every other day, all the volumes can be processed once over the course of two days. To illustrate this example, assume that the user has 200 volumes to migrate and only 105 volumes can be processed per day.

1. The user adds one day to the minimum time between processing.

```
   Current minimum processing interval  =  14 hours
 + Processing interval                  =  24 hours
   ______________________________________________

   New minimum processing interval      =  38 hours
```

   The hex values used to specify the minimum processing intervals for DFSMShsm automatic functions are expresses in seconds. An example of these values follows:

```
X'00000708' =   1,800 seconds = 30 minutes
X'0000C4E0' =  50,400 seconds = 14 hours
X'00021560' = 136,800 seconds = 38 hours
```

   The following patch defines the minimum processing interval for space management of SMS volumes to be 38 hours:

```
PATCH .MCVT.+414 X'00021560' VER(.MCVT.+414 X'0000C4E0')
```

2. Day one: Volumes 1 through 105 get processed before the stop time causes migration to stop.
3. Day two: Volumes 1 through 105 are viewed as having already been processed "today" (within the past 38 hours), so volumes 106 through 200 get processed.

Processing interval considerations for automatic primary space management: If you change the frequency of a processing interval, be aware of the following:

- Changing the frequency of the processing interval for a function applies to all occurrences of that function. You cannot use the processing interval to allow processing of one group of volumes every day and at the same time process another group of volumes every other day.
- If the change of frequency is meant to be temporary, such as for testing purposes, you need to use the PATCH command to change the frequency back to its original value when you are finished testing. To accomplish this use the PATCH command in the following way:

  For SMS volumes:

```
PATCH .MCVT.+414 X'0000C4E0' /* Reset the SMS automatic primary */
                             /* space management processing     */
                             /* interval to its original 14     */
                             /* hour day.                       */
```

  For non-SMS volumes:

```
PATCH .MCVT.+488 X'0000C4E0' /* Reset the non-SMS automatic     */
                             /* primary space management        */
                             /* processing interval to its      */
                             /* original 14 hour day.           */
```

Reasons why automatic primary space management can appear not to be running: If an automatic function does not start when you think it should, there are many conditions that can be the cause. They include but are not limited to the following:

1. The function is held or DFSMShsm is in EMERGENCY mode.
2. Today is a "N" day in the automatic processing cycle.
3. The function is already running.
4. The function appears to have already run today. This means that the window is defined in a way such that the present time is a new day in the automatic processing cycle. For automatic primary space management and automatic dump, the planned start time must be later than the last actual completion time.
5. The operator's permission reply is outstanding.

Running automatic backup multiple times a day

A consideration that applies only to backup is related to the phases of backup. See *z/OS DFSMShsm Storage Administration* for a detailed discussion of these phases. To briefly summarize, backup is divided into four phases:

- Phase 1: Backing up the control data sets
- Phase 2: Moving backup versions
- Phase 3: Backing up migrated data sets
- Phase 4: Backing up DFSMShsm-managed volumes with the automatic backup attribute

In multiple DFSMShsm-host environment configurations, only a primary host performs the first three phases. The first three phases are performed as successive phases of a single task.

Once you have determined the phases of backup you wish to perform, you need to:

1. Apply the correct patches for that phase:

   Rule: You must apply the BCR patches *after* the last time that autobackup runs. The MCVT patches will remain until DFSMShsm is restarted.

   The information you need to select and apply the correct patch is presented in the sections that follow.
2. Issue the SETSYS AUTOBACKUPSTART command to define a new automatic backup window.

   Automatic backup starts immediately if the current time is in the new start window. If the current time is not in the new start window, automatic backup starts automatically when the planned start time occurs.

Phase 1: Backing up the control data sets: If automatic backup has run to completion and you want to start automatic backup functions pertaining to any phase, you must issue the following command:

```
PATCH  .BCR.+50 X'00000000'  /* Allows all phases of        */
                             /* automatic backup to be      */
                             /* considered.                 */
```

Note: Issue the above patch after each completion of an autobackup cycle. When AUTOBACKUP completes, the date of completion is placed in the BCR at +X'50'.

Furthermore, if only the preceding patch is used in the primary host and less than 14 hours have elapsed, only the control data sets can be backed up. Understand that the use of this PATCH command on the primary host enables CDS backup. No backup of CDSs can occur unless you explicitly request that a primary host perform control data set backup.

If only the preceding patch is used in any host and 14 hours or more has elapsed since volumes were last backed up, those DFSMShsm-managed volumes with the automatic backup attribute will be backed up. If the elapsed time is less than 14 hours, the patch associated with phase 4 can be used in conjunction with the patch above to back up these same DFSMShsm-managed volumes.

Phases 2 and 3: Moving backup versions and backing up migrated data sets: If automatic backup has run to completion and you want to start automatic backup functions pertaining to phases 2 and 3, you must issue the following command:

```
PATCH  .BCR.+5C X'00000000'  /* Move backup versions created  */
                             /* by the BACKDS command and     */
                             /* back up migrated data sets.   */
```

If the preceding patch is used in a primary host in conjunction with the patch associated with phase 1, backup versions are moved and migrated data sets are backed up.

Phase 4: Backing up DFSMShsm-managed volumes with the automatic backup attribute: If automatic backup has run to completion and you want to start automatic backup functions pertaining to phase 4, you must use the patch for phase 1. The phase 1 patch enables automatic backup if the volumes have not been processed in the last 14 hours. If the volumes have been processed in the last 14 hours and you want to process them now, you must additionally issue one or both of the following commands:

- For an SMS volume:

```
PATCH  .MCVT.+418 X'00000000' /* Specify a new minimum       */
                              /* processing interval for SMS  */
                              /* volumes.                     */
```

- For a non-SMS volume:

```
PATCH  .MCVT.+48C X'00000000' /* Specify a new minimum       */
                              /* processing interval for      */
                              /* non-SMS volumes.             */
```

Processing interval considerations for automatic backup: If the change of frequency is meant to be temporary, such as for testing purposes, use the PATCH command to reset the frequency to its original value when testing is over. To accomplish this, use the PATCH command in the following manner:

- For SMS volumes:

```
PATCH .MCVT.+418 X'0000C4E0' /* Reset the SMS automatic backup  */
                             /* processing interval to its      */
                             /* original 14 hour day.           */
```

- For non-SMS volumes:

```
PATCH .MCVT.+48C X'0000C4E0' /* Reset the non-SMS automatic     */
                             /* backup processing interval to   */
                             /* its original 14 hour day.       */
```

Reasons why automatic backup can appear not to be running: If an automatic function does not start when you think it should, there are many conditions that can be the cause. They include but are not limited to the following:

- The function is held or DFSMShsm is in EMERGENCY mode.
- Today is a “N” day in the automatic processing cycle.
- The function is already running.
- The function appears to have already run today. This means that the window is defined in a way such that the present time is a new day in the automatic processing cycle. For automatic primary space management and automatic dump, the planned start time must be later than the last actual completion time.
- The operator’s permission reply is outstanding.

Example of automatically backing up the same set of SMS volumes twice a day with two different hosts: This example uses automatic backup in a multiple DFSMShsm-host environment to back up the same set of SMS volumes twice a day with two different hosts. The example shows how each of the two hosts, by using a host-specific automatic backup window and by adjusting the minimum processing interval, can back up the SMS volumes more often than the normal minimum 14-hour processing interval.

- Host 1

```
AUTOBACKUPSTART(0400 0415 0800) /* Automatic backup will begin */
                                /* at 4:00 a.m. with a latest  */
                                /* start time of 4:15 a.m. and */
                                /* end at 8:00 a.m.            */

PATCH .MCVT.+418 X'00006270'    /* Adjust the minimum time for */
                                /* automatic backup from 14    */
                                /* hours to 7 hours.           */
```

Because host 2 finishes backup at 8 p.m. and host 1 begins backup at 4 a.m., approximately eight hours elapse between backups. Therefore, host 1 needs to use a patch to shorten the 14-hour minimum processing interval to less than eight hours.

- Host 2

```
AUTOBACKUPSTART(1600 1615 2000) /* Automatic backup will begin */
                                /* at 4:00 p.m. with a latest  */
                                /* start time of 4:15 p.m. and */
                                /* end at 8:00 p.m.            */

PATCH .MCVT.+418 X'00006270'    /* Adjust the minimum time for */
                                /* automatic backup from 14    */
                                /* hours to 7 hours            */
```

Note: The hex value used to specify the minimum processing interval is expressed in seconds:

```
X'00006270' = 25,200 seconds = 7 hours
```

Because host 1 finishes backup at 8 a.m. and host 2 begins backup at 4 p.m., approximately eight hours elapse between backups. Therefore, host 2 needs to use a patch to shorten the 14-hour minimum processing interval to less than eight hours.

Running automatic dump multiple times a day in a test environment

If automatic dump has been performed on a set of volumes and if you want to perform automatic dump again on the same set of volumes, perform the following steps:

1. Issue one or both of the following PATCH commands:
   - For an SMS volume:

     ```
     PATCH  .MCVT.+41C X'00000000' /* Reset the 14 hour day default */
                                   /* for SMS automatic dump        */
     ```

   - For a non-SMS volume:

     ```
     PATCH  .MCVT.+490 X'00000000' /* Reset the 14 hour day default */
                                   /* for non-SMS automatic dump    */
     ```

2. Issue the SETSYS AUTODUMPSTART command to define a new start window. Specify a process window that has a planned start time after the time that automatic dump last ended. You can determine the time that automatic dump last ran from the dump activity log.

The SMS patches listed for automatic dump are not necessary if all dump activity is of copy pools. Automatic dump does not restrict copy pools to the 14 hour minimum for dump processing.

Automatic dump starts immediately if the current time is in the new start window. If the current time is not in the new start window, automatic dump starts automatically when the planned start time occurs.

Processing interval considerations for automatic dump: If the change of frequency is meant to be temporary, such as for testing purposes, you need to use the PATCH command to reset the frequency to its original value when done testing. To accomplish this, use the PATCH command in the following manner:

- For SMS volumes:

```
PATCH .MCVT.+41C X'0000C4E0'     /* Reset the SMS automatic dump   */
                                 /* processing interval to its     */
                                 /* original 14 hour day           */
```

- For non-SMS volumes:

```
PATCH .MCVT.+490 X'0000C4E0'       /* Reset the non-SMS automatic    */
                                   /* processing interval to its     */
                                   /* original 14 hour day           */
```

Reasons why automatic dump can appear not to be running: If an automatic function does not start when you think it should, there are many conditions that can be the cause. They include but are not limited to the following:

- The function is held or DFSMShsm is in EMERGENCY mode.
- Today is a "N" day in the automatic processing cycle.
- The function is already running.
- The function appears to have already run today. This means that the window is defined in a way such that the present time is a new day in the automatic processing cycle. For automatic primary space management and automatic dump, the planned start time must be later than the last actual completion time.
- The operator's permission reply is outstanding.

Changing the frequency of running interval migration

DFSMShsm checks the occupied space on each managed DASD volume to determine the need for migration at certain times during the day. Thus, the "interval" for such migration is actually the interval at which DFSMShsm checks space. To change the frequency for interval migration requires changing one or more of the following values that are used to control how often DFSMShsm checks occupied space:

- For a primary host (or a non-primary host that cannot determine the status of the primary host), the number of minutes (default of 90) that is added to the end time of the current space check, then truncated to the hour, to determine the time for the next space check on this host.
- For a non-primary host, the number of minutes (default of 5) that this host delays after the primary host has completed or was expected to complete, its space check.
- The minimum number of minutes (default of 30) between successive migrations of the same SMS-managed volume.
- The minimum number of minutes (default of 30) between successive migrations of the same non-SMS-managed volume.

Note:

1. Interval migration patch values are ignored for SMS-managed volumes when on-demand migration is used.
2. DFSMShsm will always use the default interval migration frequency to schedule the first space check; even if a patch is used in the startup PARMLIB member.

Related patches

- "Making the interval less frequent than one hour"
- "Making the interval more frequent than one hour"

Making the interval less frequent than one hour

To calculate the value to use in a patch of the MGCB, start with the desired number of hours for your interval, multiply by 60, and then add 30. For example, if you want the interval to be two hours, the value is (2*60)+30 = 150. Issue the following PATCH command:

```
PATCH .MGCB.+60 X'0096'    /* 150 minutes */
```

Making the interval more frequent than one hour

If you want the space check to occur once per hour in each of *n* hosts, but at a different time after the hour in each host, issue the following patch. Set one field to the number of minutes after the hour when you want the space check to start, and set another field so that the sum of the two is 90.

To start interval migration twice per hour requires two hosts. Issue the following patches to start space check on the primary host on the hour and start space check on the non-primary host at 30 minutes after the hour:

```
Primary:      PATCH .MGCB.+60 X'005A'  /* 90 minutes - default  */

Non-primary:  PATCH .MGCB.+62 X'001E'  /* 30 minutes after the  */
                                       /* hour is the time to   */
                                       /* do space check        */

              PATCH .MGCB.+60 X'003C'  /* 60 more minutes, to   */
                                       /* take us to the middle */
                                       /* of the next hour      */
```

Issue the following patches to set the minimum number of minutes between successive migrations of the same volume to one-half the hourly value:

```
Primary:   PATCH .MCVT.+422 X'000F'   /* 15 min, SMS      */
           PATCH .MCVT.+496 X'000F'   /* 15 min, non-SMS  */

Non-primary:  PATCH .MCVT.+422 X'000F' /* 15 min, SMS     */
              PATCH .MCVT.+496 X'000F' /* 15 min, non-SMS */
```

If you choose not to use a primary host for interval migration, issue the following patch to substitute a non-primary host by setting its space check time to one minute after the hour:

```
Non-primary 2:  PATCH .MGCB.+62 X'0001'  /* 1 minute after the  */
                                         /* hour is the time to */
                                         /* do space check      */
```

Starting interval migration three times per hour requires three hosts. Issue the following patches to start space check on the primary host on the hour, on the first non-primary host at 20 minutes after the hour, and on the second non-primary host at 40 minutes after the hour:

```
Primary:      PATCH .MGCB.+60 X'005A'    /* 90 minutes - default  */

Non-primary 1:  PATCH .MGCB.+62 X'0014'  /* 20 minutes after the  */
                                         /* hour is the time to   */
                                         /* do space check        */

                PATCH .MGCB.+60 X'0046'  /* 70 more minutes, to   */
                                         /* take us to the middle */
                                         /* of the next hour      */

 Non-primary 2:  PATCH .MGCB.+62 X'0028' /* 40 minutes after the  */
                                         /* hour is the time to   */
                                         /* do space check        */

                PATCH .MGCB.+60 X'0032'  /* 50 more minutes, to   */
                                         /* take us to the middle */
                                         /* of the next hour      */
```

Issue the following patches to set the minimum number of minutes between successive migrations of the same volume to one-third the hourly value:

```
Primary:   PATCH .MCVT.+422 X'000A'      /* 10 min, SMS      */
           PATCH .MCVT.+496 X'000A'      /* 10 min, non-SMS  */

Non-primary 1:  PATCH .MCVT.+422 X'000A' /* 10 min, SMS      */
                PATCH .MCVT.+496 X'000A' /* 10 min, non-SMS  */

Non-primary 2:  PATCH .MCVT.+422 X'000A' /* 10 min, SMS      */
                PATCH .MCVT.+496 X'000A' /* 10 min, non-SMS  */
```

If you choose not to use a primary host for interval migration, issue the following patch to substitute a non-primary host by setting its space check time to one minute after the hour:

```
Non-primary 3:  PATCH .MGCB.+62 X'0001'  /* 1 minute after the hour */
                                         /* hour is the time to     */
                                         /* do space check          */
```

Changing the frequency of running on-demand migration again on a volume that remains at or above the high threshold

After a volume is processed using on-demand migration it can remain at or above the high threshold. This happens when there are not enough eligible data sets on the volume to be space managed. To prevent continuously processing a volume that remains above the high threshold, an on-demand migration timer is started. This timer prevents on-demand migration from running again for 24 hours (default) on all volumes that remained at or above the high threshold during its duration. When the timer expires, on-demand migration is performed again on all of the volumes that remain at or above the high threshold. A patch is provided to change the frequency at which volumes at or above the high threshold are selected for on-demand migration again.

For example, you can use the following PATCH command to change the frequency to 48 hours:

```
PATCH .MGCB.+138 X'0002A300' /* 48 hours in seconds */
```

To restore the default frequency issue:

```
PATCH .MGCB.+138 X'00015180' /* 24 hours in seconds - default value */
```

Note: The value specified in the patch is the number of hours expressed in hexadecimal seconds.

Reducing enqueue times on the GDG base or on ARCENQG and the fully qualified GDS name

Users can choose whether increasing throughput via migration scratch queue (MSQ) processing is more important than reducing enqueue times for the GDG base or for ARCENQG and the fully qualified GDS name.

If you issue the following PATCH command:

```
PATCH .MGCB.+ED BITS(....1...)
```

then reducing the enqueue times for the GDG base will be selected and MSQ processing for the GDS will be suspended. Otherwise, DFSMShsm will operate normally.

If you issue the following PATCH command:

```
PATCH .MGCB.+EF BITS(....1...)
```

then reducing the enqueue times for ARCENQG and the fully qualified GDS name will be selected and MSQ processing for the GDS will be suspended. Otherwise, DFSMShsm will operate normally.

Modifying the migration queue limit value

The migration queue limit value can be modified to something other than the default of 50 000. This is recommended only if your installation is receiving too many unwanted ARC0535I messages.

To change the default value to a higher value, in this example 100 000 (X'186A0'), issue:

```
PATCH .MGCB.+100 X'00186A0' VERIFY(.MGCB.+100 X'0000C350')
```

To return to the default value of 50 000 (X'C350'), issue:

```
PATCH .MGCB.+100 X'0000C350' VERIFY(.MGCB.+100 X'00186A0')
```

Note:

1. Take care in increasing the migration queue limit to a value that is reasonable, based upon the number of data sets which reside on your volumes. Increasing the migration queue limit to a value that is too high may cause ABEND878s.

2. If you do intend on increasing the migration queue limit, then you should also consider decreasing the amount of migration tasks to avoid a drastic increase in the amount of virtual memory used.
3. The patch values are reset to the default after DFSMShsm is restarted. In order to modify it again, you will need to reissue the patch command, or else you can add the patch command to the ARCCMD*xx* parmlib member that DFSMShsm uses during startup.

Changing the default tape data set names that DFSMShsm uses for tape copy and full volume dump

In releases of DFHSM prior to Version 2 Release 6.0, the default names for tape copy and full volume dump tape data sets were different from the default names used in DFHSM Version 2 Release 6.0 and in subsequent DFSMShsm releases.

Some sites have used and continue to use naming conventions established prior to DFHSM Version 2 Release 6.0. As a convenience, this patch is provided to eliminate changing naming conventions that have been established with previous releases of DFHSM.

Default tape data set names for DFHSM Version 2 Release 6.0

The default tape data set names that DFHSM Version 2 Release 6.0 uses for tape copy and full volume dump are:

Kind of Tape	Default Tape Data Set Name
Tape copy of a migration tape	*prefix*.COPY.HMIGTAPE.DATASET
Tape copy of a backup tape	*prefix*.COPY.BACKTAPE.DATASET
Full volume dump tape	*prefix*.DMP.dclass.Vvolser.Dyyddd. Tssmmhh

Default tape data set names for DFHSM releases prior to Version 2 Release 6.0

The following are default tape data set names that DFHSM used for tape copy and full volume dump, prior to DFHSM: Version 2 Release 6.0:

Kind of Tape	Default Tape Data Set Name
Tape copy of a migration tape	*prefix*.HMIGTAPE.DATASET
Tape copy of a backup tape	*prefix*.BACKTAPE.DATASET
Full volume dump tape	*prefix*.DMP.Tssmmhh.dclass.Dyyddd. Vvolser

If you want DFSMShsm to use a naming convention that is consistent with releases of DFHSM, you can issue the following patch:

```
PATCH  .MCVT.+284 BITS(.0......)       /* Change to default tape copy    */
                                       /* name for releases of DFHSM     */
                                       /* prior to Version 2 Release 6.0.*/

       PATCH  .MCVT.+284 BITS(0.......) /* Change to default full volume  */
                                       /* dump tape name for releases of */
                                       /* DFHSM prior to Version 2       */
                                       /* Release 6.0.                   */
```

Preventing interactive TSO users from being placed in a wait state during a data set recall

When an interactive-time-share (TSO) user refers to a migrated data set, DFSMShsm recalls the data set to the user’s level-0 volume. During the data-set recall, the user is put into a wait state. To allow the recall to continue and yet return to an interactive session, the user can press the attention key.

After the user "attentions out" of the recall, the screen is left with meaningless data. To clear the screen of meaningless data and to return to the user's TSO session, it is necessary to press the PA2 key.

Many users do not know that pressing the PA2 key clears the screen of meaningless data. A site can eliminate the need for users to know about the PA2 key by issuing the following PATCH command. Issue the following PATCH command to change the default for wait states to a default of no wait. When the PATCH command is in effect, DFSMShsm schedules the recall, and issues message ARCI020I to let the user know that the data set is being recalled and that the data set may be accessed after the recall completes. If the data set recall is generated as a result of accessing the data set through a TSO panel, ARCI020I is followed by another message indicating that the locate failed.

```
PATCH  .MCVT.+52 bits(..1.....)  /* Alter the default for tape    */
                                 /* data set recalls from a wait  */
                                 /* state to a no-wait state      */



 PATCH  .MCVT.+52 bits(...1....)  /* Alter the default for DASD    */
                                  /* data set recalls from a wait  */
                                  /* state to a no-wait state      */
```

If you want the no-wait default to be effective at startup, ensure that you include the previous PATCH command in the startup procedure. If the PATCH commands are entered after DFSMShsm startup, they are not effective for the first recall.

After a recall is generated because of the UNCATALOG function from TSO option 3.2, a second recall is automatically generated by option 3.2 after the first recall is intercepted by DFSMShsm. The second recall is not processed until after the first recall has completed. Consequently, the second recall fails because the data set has already been recalled.

Preventing ABARS ABACKUP processing from creating an extra tape volume for the instruction data set and activity log files

If you direct your ABACKUP activity log to DASD or if you specify an instruction data set name in the aggregate group definition, ABARS invokes DFSMSdss to dump those data sets to a separate tape volume from the control file and data files. This causes ABACKUP processing to require a minimum of three tape volumes. Since you may not wish to use a third tape volume, DFSMShsm allows you to control whether or not this tape is created.

You can use the DFSMShsm PATCH command to modify the ABRCB control block to prevent DFSMSdss from dumping the ABACKUP activity log, the instruction data set, or both. If you modify ABRCB in this way, ABACKUP processing will not create a separate output volume for these data sets.

There are four DFSMShsm PATCH command options that control whether the activity log and instruction data set are dumped to a separate tape volume.

The following option allows ABARS to continue to dump the activity log and instruction data set to a separate tape volume.

```
PATCH .ABRCB.+1F X'00' -
```

The dump occurs if either an instruction data set name was specified in the aggregate group definition or if the ABACKUP activity log was directed to DASD when you specified SETSYS ABARSACTLOGTYPE(DASD). This is the default setting. You need to use this setting only when you are resetting the options.

The following option directs ABARS to invoke dump whenever the ABACKUP activity log is directed to DASD, without regard to the existence of an instruction data set.

```
PATCH .ABRCB.+1F X'01' -
```

The following option directs ABARS to invoke dump whenever an instruction data set name is specified in the aggregate group definition, whether or not the ABACKUP activity log was directed to DASD.

```
PATCH .ABRCB.+1F X'02' -
```

The following option directs ABARS to never invoke DFSMSdss to dump the ABACKUP activity log or instruction data set. This is the way to tell ABARS not to create a third tape output volume under any circumstances.

```
PATCH .ABRCB.+1F X'03' -
```

When you prepare for aggregate recovery at your recovery site, and you have used these PATCH commands to modify the ABRCB at your backup site, you should begin aggregate recovery with a BCDS without any ABR records. You should then issue the ARECOVER command with the PREPARE, VERIFY, or EXECUTE option to create a new ABR record. If you created an instruction data set or activity log file during ABACKUP processing, you can use the ACTIVITY or INSTRUCT parameters when you issue the ARECOVER PREPARE VERIFY or EXECUTE instructions to restore these files.

Note: This patch does not affect ABACKUP output when installations have specified SETSYS ABARSTAPES(STACK).

Preventing ABARS ABACKUP processing from including multivolume BDAM data sets

DFSMShsm ABACKUP invokes DFSMSdss to process level-0 data sets in the INCLUDE list. During ARECOVER, DFSMSdss is unable to recover multivolume BDAM data sets. To prevent ABACKUP from processing such data sets, issue the following patch:

```
PATCH .ABRCB.+A8 BITS(.....1..)
```

If this patch is set, ABACKUP bypasses all multivolume BDAM data sets and continues processing. The reason why ABACKUP bypasses *both* SMS-managed and non-SMS-managed multivolume BDAM data sets is that you could direct a non-SMS-managed multivolume BDAM data set to be SMS-managed at the ARECOVER site, and DFSMSdss would fail to recover the data set.

Note: This patch should not be used unless your installation *has* multivolume BDAM data sets in its INCLUDE list, because an extra OBTAIN is needed to identify and bypass these data sets; the OBTAIN will impact ABACKUP's performance.

Specifying the amount of time to wait for an ABARS secondary address space to initialize

When you run the ABARS startup procedure, DFSMShsm internally issues an MVS START command to start the ABARS secondary address space. If the address space does not start within a specified time, the ABARS address space is canceled and MVS frees the address space resources. The default time in which the ABARS address space must start is 5 minutes (300 seconds).

You can alter the default startup wait time with the following PATCH command:

```
PATCH .ABRCB.+494 X'nnnnnnnn' VERIFY(.ABRCB.+494 X'0000012C')
```

Note: X'nnnnnnnn' is the hexadecimal representation for the number of seconds to wait for an ABARS secondary address space to start.

Do not reduce the wait time unless you are sure that the ABARS address space can start within that time frame. If the time frame is too short, unexpected results can occur.

Patching ABARS to use NOVALIDATE when invoking DFSMSdss

The DFSMSdss VALIDATE function provides extra data integrity checking during the ABACKUP process. You should not use this patch unless there is an explicit need for down-level compatibility. A new bit in the ABRCB control block has been defined to provide this support.

The following patch instructs ABARS ABACKUP to *not use* the new VALIDATE logic, to allow consistency with down-level recovery sites:

```
PATCH .ABRCB.+82 BITS(1.......)    /* Do not use VALIDATE */
```

Patching ABARS to provide dumps whenever specific errors occur during DFSMSdss processing during ABACKUP and ARECOVER

The ABARS function provides a patchable field in the ABRCB so that a customer can specify to abend DFSMSdss when a specific DFSMSdss message is issued.

The offset to patch in the ABRCB is 1C. For example, to abend DFSMSdss when the ADR454I message is issued, you would issue the following DFHSM PATCH command:

```
PATCH .ABRCB.+1C X'F4F5F4'
```

Routing ABARS ARC6030I message to the operator console

When this patch is applied it will instruct the ABARS function to route the ARC6030I message to the operators console, as well as, any other previously requested output avenue.

Issue the following PATCH command to route the ARC6030I message to the operator console:

```
PATCH .ABRCB.+81 BITS(.....1..)
```

Filtering storage group and copy pool ARC0570I messages (return code 17 and 36)

During automatic backup, dump, and migration, SMS storage group and copy pool information is retrieved. If this information cannot be retrieved, message ARC0570I issued. Specifically:

- return code 17 is issued when storage group information cannot be retrieved.
- return code 36 is issued when copy pool information cannot be retrieved.

In an SMS environment these return codes can indicate an error. However, in a non-SMS environment these return codes do not provide meaningful information as storage groups and copy pools do not exist in a non-SMS environment.

Filtering message ARC0570I return code 17

Issue the following PATCH command to *enable* filtering of ARC0570I RC=17 messages:

```
PATCH .MCVT.+297 BITS(....1...)
```

Issue the following PATCH command to *disable* filtering of ARC0570I RC=17 messages:

```
PATCH .MCVT.+297 BITS(....0...)
```

Filtering message ARC0570I return code 36

Issue the following PATCH command to *enable* filtering of ARC0570I RC=36 messages:

```
PATCH .MCVT.+297 BITS(.....1..)
```

Issue the following PATCH command to *disable* filtering of ARC0570I RC=36 messages:

```
PATCH .MCVT.+297 BITS(.....0..)
```

Note: These return codes are not filtered when using the LIST command.

Allowing DFSMShsm to issue serialization error messages for class transitions

By default, the serialization error messages from DFSMShsm are suppressed. To allow issuance of the ARC0734I messages when serialization errors occur during class transition processing, issue the following PATCH command:

```
PATCH .MGCB.+EF BITS(...1....)
```

Enabling ARC1901I messages to go to the operator console

To specify that ARC1901I messages be issued on the operator console and the log, issue the following PATCH command:

```
PATCH .MGCB.+115 BITS(..1.....)
```

To specify that ARC1901I messages be issued to the log only, issue the following PATCH command:

```
PATCH .MGCB.+115 BITS(..0.....)
```

By default, ARC1901I messages go to the operator console and the log.

Changing the notification limit percentage value to issue ARC1901I messages

The ARC1901I messages are issued for every *x*% of the notification limit value, where *x* is a decimal number from 1 to 100. The default value is 20%.

For example, if the notification limit is 100 and the percentage is 20, then ARC1901I will be issued when the number of queued volumes is 100, 120, 140, and so on.

You can change the notification limit percentage value with the following PATCH command:

```
PATCH .MGCB.+154 X'000000nn'
```

where *nn* is the hexadecimal representation for the notification limit percentage value.

For example, to set the percentage value to 50%, issue the following PATCH command:

```
PATCH .MGCB.+154 X'00000032'
```

Patching to prevent ABARS from automatically cleaning up residual versions of ABACKUP output files

If a failure occurs during a previous ABACKUP for an aggregate, it is possible that a residual ABACKUP output file is left over and was not cleaned up (usually due to an error condition). The ABACKUP processing normally detects this and deletes the residual file. If the ABACKUP processing detects an existing output files for the version currently being created, the ABACKUP function deletes the files.

To prevent the ABACKUP function from deleting the output files, issue the following PATCH command:

```
PATCH .ABRCB.+82 BITS(.1......)
```

If the patch is set on, ABACKUP processing issues the ARC6165E message, ABACKUP processing fails, and the user will have to manually delete or rename the files.

Enabling the serialization of ML2 data sets between RECYCLE and ABACKUP

The ABACKUP function locates data sets, including data sets on ML2 tapes, before backing them up. Before ABACKUP copies the data sets, RECYCLE processing may be invoked to process ML2 tapes. If ABACKUP tries to access data sets while RECYCLE is moving them from an ML2 tape, ABACKUP fails.

To prevent RECYCLE from moving a data set that ABACKUP is processing, issue the following PATCH command:

```
PATCH .MCVT.+195 BITS(......1.)
```

This patch turns ON a new bit, which causes RECYCLE to issue an enqueue request on the same resource that ABACKUP is processing. That is, RECYCLE issues an enqueue on the resource ARCDSN/*dsname*, where *dsname* is the name of the data set on ML2. Enqueue requests result in one of the following conditions:

- If the enqueue fails, RECYCLE processing does not move the data set. If any data sets were skipped, a subsequent recycle of the volume will be necessary to move them.
- If the enqueue succeeds, RECYCLE de-serializes the resource after the data set is moved.

Note: Because these enqueues may negatively impact RECYCLE performance, they are performed only if you issue the PATCH command for any DFSMShsm on which RECYCLE runs.

Changing the default number of recall retries for a data set residing on a volume in use by RECYCLE or TAPECOPY processing

RECYCLE— When a DFSMShsm user needs a data set residing on a volume that is being processed by RECYCLE processing, the recall function requests the volume and then retries the recall request approximately every two minutes for up to 30 minutes (15 retries). After 15 retries, DFSMShsm issues message ARC0380A, which requires the operator to respond with a WAIT, CANCEL, or MOUNT command.

TAPECOPY— By default, when a DFSMShsm user needs a data set that resides on a volume that is being read by TAPECOPY processing, that data set cannot be recalled until TAPECOPY processing has completed. By default, DFSMShsm retries the recall request approximately every two minutes for up to 30 minutes (15 retries). After 15 retries, DFSMShsm issues message ARC0380A, which requires the operator to respond with a WAIT, CANCEL, or MOUNT command. For more information about enabling the takeaway function during TAPECOPY, see “Enabling the takeaway function during TAPECOPY processing” on page 358.

For both RECYCLE and TAPECOPY, you can increase or decrease the number of recall request retries by modifying a patchable field in the MCVT. For example, the following patch resets the number of recall request retries to zero.

```
PATCH .MCVT.+315 X'00'    /*Fail recall if tape is in use */
                                    /* by recycle or tapecopy       */
```

Although 15 is the default for the number of times a recall is retried when the volume is in use by RECYCLE or TAPECOPY (retries occur approximately every 2 minutes for a total of 30 minutes), you can use the following patch to change the number of RECALL request retries to, for example, 30 for a total of one hour.

```
PATCH .MCVT.+315 X'1E'     /* Retry the recall 30 times if */
                                     /* the volume is in use by       */
                                     /* recycle or tapecopy           */
```

Changing the default number of buffers that DFSMShsm uses to back up and migrate data sets

DFSMShsm processes user data sets during automatic backup processing with only one buffer because some sites with heavy processing loads have experienced storage constraints with multiple buffers. Depending on the DFSMShsm configuration at a site, the number of buffers for automatic backup processing may be increased without any storage constraints. You can control the number of buffers that DFSMShsm uses for backup and migration with the following PATCH command:

```
 PATCH .MCVT.+391 X'nn'VERIFY(.MCVT.+391 X'00')  /* Change the default number*/
                                                 /* of buffers for backup and*/
                                                 /* migration of data sets   */
```

One buffer is the default (specified as *nn*=X'01'), but you can increase the number of buffers (*nn*) to five (X'05').

Changing the compaction-ratio estimate for data written to tape

When DFSMShsm is migrating or backing up a data set to a tape device supporting hardware compaction, if the data set is not already flagged as compressed, DFSMShsm assumes a compaction of 2.5 (25/10) will occur, in estimating whether the data set will fit entirely on the currently mounted cartridge.

If a high percentage of your data has attributes such that the assumed compaction ratio is not appropriate, you can change the ratio used, by issuing a PATCH command to specify the numerator of a ratio having a denominator of 10. You can change the ratio for migration or backup tape or both. For example:

```
PATCH .MCVT.+4B6 X'0F' /* Migration - ratio is 15/10 or 1.5 */
```

```
PATCH .MCVT.+4B7 X'14' /* Backup - ratio is 20/10 or 2.0 */
```

Enabling the takeaway function during TAPECOPY processing

By default, when a DFSMShsm user needs a data set that resides on a volume that is being read by TAPECOPY processing, that data set cannot be recalled or backed up by ABACKUP function until TAPECOPY processing has completed. During TAPECOPY processing, recalls are tried approximately every two minutes for 30 minutes (15 retries). After 15 retries, DFSMShsm issues message ARC0380A, which requires the operator to respond with WAIT, CANCEL, or MOUNT commands.

DFSMShsm retries the ABACKUP request every twenty seconds for up to 30 minutes (90 retries). After 90 retries, DFSMShsm issues message ARC6254A, which requires the operator to respond with Y (retry again) or N (fail the ABACKUP).

If takeaway from TAPECOPY is enabled, recall tries every two minutes for 15 minutes, and ABACKUP tries every 20 seconds for 15 minutes. Recall or ABACKUP then requests that TAPECOPY end on the needed tape.

To enable the takeaway tape function during TAPECOPY processing, issue the following PATCH command:

```
PATCH .MCVT.+53 BITS (.....1..)
```

This patch enables RECALL to request that TAPECOPY processing end on the needed tape, but does not do so unless 15 minutes have passed since the first attempt of the recall. DFSMShsm no longer issues message ARC0380A.

You can also change the delay time of 15 minutes. For example, to change the delay time to 20 minutes, issue the following PATCH command:

```
PATCH .MCVT.+3CA X'0014'
```

You can also change the ABACKUP delay time of 15 minutes. For example, to change the delay time to 25 minutes, issue the following command:

```
PATCH .ABRCB.+28 XX'0019'
```

Changing the delay by recall before taking away a needed ML2 tape from ABACKUP

When a WAIT-type recall needs an ML2 tape currently in use by ABACKUP, recall keeps retrying the access to the tape. If ABACKUP is still using the tape after a delay of ten minutes since recall first found the tape in use, recall signals ABACKUP to give up the tape.

You can change that delay time (in minutes) by issuing the following PATCH command:

```
PATCH .MCVT.+49E X'000F'        /* Have wait-type recall   */
                                /* delay 15 minutes before */
                                /* taking away a tape from */
                                /* ABACKUP                  */
```

By making this a large number (maximum of X'FFFF'), you can prevent recall from taking away a tape from ABACKUP.

The number of passes that ABACKUP can make to delay its uses between passes are patchable as described in “Changing the amount of time ABACKUP waits for an ML2 volume to become available” on page 361.

Disabling delete-if-backed-up (DBU) processing for SMS data sets

Before SMS-managed data sets are expired, a check is made to ensure that the data set has a backup copy. Some sites want to override the requirement that SMS-managed data sets have a backup copy before they are expired. The following patch provides this capability:

```
PATCH .MCVT.+431 BITS(......1.)    /* Override the requirement    */
                                   /* that SMS-managed data sets  */
                                   /* have a backup copy before   */
                                   /* they can be expired         */
```

Requesting the message issued for SETSYS TAPEOUTPUTPROMPT processing be WTOR instead of the default WTO

You can apply a patch that allows DFSMShsm to issue a WTOR message for SETSYS TAPEOUTPUTPROMPT processing instead of the current WTO message. The WTOR message is ARC0332R and the default WTO message is ARC0332A. Message ARC0332R is satisfied when the tape is mounted and opened, and the operator replies 'Y' to the outstanding WTOR. This gives the operator a chance to ensure that the correct type of tape is mounted, even if ACL/ICL devices are being used. Processing will not continue until 'Y' is entered. The patch to request a WTOR is:

```
PATCH .MCVT.+4C3 BITS(.......1)
```

The patch to revert to the WTO is:

```
PATCH .MCVT.+4C3 BITS(.......0)
```

Removing ACL as a condition for D/T3480 esoteric unit name translation

D/T3480 tape devices with automatic cartridge loaders (ACLs) have esoteric unit name translation. D/T3480s without ACLs will translate the esoteric unit name with the following PATCH command:

```
PATCH .MCVT.+4C0 BITS(...1....)
```

The patch must be in the DFSMShsm parmlib before the SETSYS USERUNITTABLE command. The output devices associated with the esoteric unit name must be online when the SETSYS USERUNITTABLE command is specified.

Restricting non-SMS ML2 tape volume table tape selection to the SETSYS unit name of a function

In a non-SMS tape environment, if you only want ML2 volumes with the same stored unit name as the SETSYS unit name of a function to be considered for output, issue the following PATCH command:

```
PATCH .MCVT.+1BC BITS(...1....)
```

When using this patch, ML2 tapes in the ML2 tape volume table (TVT) are considered for output when the MCVUNIT field matches the SETSYS unit name of the function being processed.

This patch can be useful when the tape hardware does not follow IBM conventions for reporting to DFSMShsm. For instance, if the migration function uses one type of non-IBM tape hardware and the recycle function uses another type of non-IBM tape hardware, both reporting as 3590B, DFSMShsm tape selection logic cannot distinguish between the two different tape hardware types.

To distinguish between the two tape hardware types, esoteric unit names can be used and the SETSYS USERUNITTABLE command can be used to define the distinct unit names. For example, the following commands establish the esoteric unit names and distinguish which esoteric unit name is used for the migration and recycle functions:

```
SETSYS USERUNITTABLE(esoteric1:esoteric1,esoteric2:esoteric2)
SETSYS TAPEMIGRATION(ML2TAPE(TAPE(esoteric1)))
SETSYS RECYCLEOUTPUT(MIGRATION(esoteric2))
```

The esoteric unit names on the right side of the colon in the SETSYS USERUNITTABLE command are stored in the CDS volume records. These unit names are used to match a unit name with the SETSYS unit name of a specific function. If they are omitted from the command, the generic equivalent of the esoteric is stored instead. Because the generic equivalent might not match the SETSYS unit name, the tape might be rejected when this patch is in effect.

Changing the amount of time ABACKUP waits for an ML2 volume to become available

When ABACKUP processes a data set on ML2 tape, it updates the primary and migration volume record (MCV) of the volume being processed to indicate that it is in use by ABACKUP. If the data set spans volumes, the MCVs of the first and last volumes are updated. If a tape volume is already in use by another function, ABACKUP waits up to 30 minutes for the volume to become available as the default. If after waiting 30 minutes, the volume still remains unavailable, message ARC6254A prompts the operator to continue the wait or to cancel the operation. The end of this wait means that the MCV has become unmarked as being in use by the other function; no physical tape allocation is attempted until after the MCV has been updated.

You can alter the default wait time of 30 minutes by entering the following PATCH command:

```
PATCH .ABRCB.+4A2 X'nnnn' VERIFY(.ABRCB.+4A2 X'005A')
```

Note: X'nnnn' is the hexadecimal representation of the number of 20-second intervals to be tried. The default decimal value is 90, which produces a 30-minute wait.

If ABACKUP finds, while backing up migrated data sets, that an ML2 tape is in use by another ABACKUP task, it temporarily skips backing up from that tape and continues backing up from any other needed ML2 tapes that are not in use.

After all ML2 tapes have been processed or found to be in use, another pass can occur if any ML2 tapes were skipped. After a tape was skipped, and if no other data sets were backed up from tape in the current pass, ABACKUP delays for five minutes waiting for the other ABACKUP commands to complete using the ML2 tapes in contention. ABACKUP then retries accessing the data sets on the ML2 tapes that were skipped during the previous pass. If at least one ML2 tape is still in use, ABACKUP retries again, starting another pass.

ABACKUP retries a maximum of nine times, for a possible total of ten attempts.

If at the end of nine retries there is still at least one ML2 tape still in use by another ABACKUP command, ABACKUP issues an ARC6254A message for each such volume. If any response is N (No), ABACKUP processing fails with an ARC6259E message without any additional retries. If all the responses are Y (Yes), ABACKUP writes one ARC6255I message and retries a maximum of another nine times. If at the end of the second nine retries, there is still at least one ML2 tape in use, the ABACKUP processing fails with an ARC6261E message.

To patch the between-pass delay value, issue the following command:

```
PATCH .ABRCB.+2A X'0008' /* Delay by ABACKUP for 8 minutes */
                         /* between passes for another     */
                         /* ABACKUP to finish with an ML2  */
                         /* needed by both                 */
```

To patch the number of retry loops, issue the following command:

```
PATCH .ABRCB.+38 X'04'   /* Make 4 retry loops for ABACKUP */
                         /* needing ML2 tape(s) in use by  */
                         /* other ABACKUP(s)               */
```

Changing the amount of time an ABACKUP or ARECOVER waits for a resource in use by another task

When an aggregate backup or recovery fails to allocate a volume or data set, it will retry the allocation for up to 30 minutes before issuing message ARC6083A. This message prompts the user to respond to the message with a CANCEL (cancel the operation) or a WAIT (continue to wait for the resource).

You can alter the default wait time of 30 minutes by entering the following PATCH command:

```
PATCH .ABRCB.+4C4 X'nnnn' VERIFY(.ABRCB.+4C4 X'005A')
```

Note: X'nnnn' is the hexadecimal representation of the number of 20-second intervals to be tried. The default decimal value is 90, which produces a 30-minute wait.

Preventing deadlocks during volume dumps

DUMP processing holds the VTOC resource while dumping a volume. Whenever DUMP processing reaches the EOV, it must get the TIOT resource in order to access a new tape. However if a request is issued to allocate a data set on the DASD being dumped, the request holds the TIOT resource and requests the VTOC resource instead. This inverse ordering of resource acquisition can result in a function deadlock.

To activate the early VTOC release functional change, put the following PATCH command into your DFSMShsm startup procedure:

```
PATCH .MCVT.+3C3 BITS(.....1..)
```

You can also activate early VTOC release by invoking the ADRUENQ installation exit. See *z/OS DFSMS Installation Exits* for more information on the ADRUENQ installation exit.

Modifying the number of elapsed days for a checkpointed data set

You can modify the number of days that must have elapsed since the date last referenced in order for a checkpointed data set (DS1CPOIT—with or without DS1DSGU) to be eligible for migration.

To modify the number of days that must have elapsed, issue the following PATCH command:

```
PATCH .MGCB.+70 X'nn'VERIFY(.MGCB.+70 X'05')
```

Note: X'nn' is the hexadecimal representation for the number of days. The default is five.

Running concurrent multiple recycles within a single GRSplex

Any customer that recycles the same category of DFSMShsm tapes at the same time on different hosts within a GRSplex will encounter failed requests.

Attention: Use the following patches to avoid recycle conflicts between two HSMplexes that use the RNAMEDSN=NO translation method. If you are using RNAMEDSN=YES on either or both, the following patches are unnecessary.

If your environment has a single GRS-type serialization ring that includes more than one HSMplex, you may want to use different resource names (Rnames) for the recycle tape category, which allows multiple HSMplexes to recycle the same category concurrently. You can patch the Rnames to represent different resources. Place the patches in the DFSMShsm startup procedure.

For example, the following PATCH commands illustrate how you could change the RNAMEs to reflect HSMPLEX 1 ('H1').

```
PATCH .YGCB.+14 'RCYH1-L2' VER(.YGCB.+14 'RECYC-L2')
PATCH .YGCB.+1C 'RCYH1-SP' VER(.YGCB.+1C 'RECYC-SP')
PATCH .YGCB.+24 'RCYH1-DA' VER(.YGCB.+24 'RECYC-DA')
```

This change does not require that you change the QNAME.

To provide the needed protection, make sure that you use the same resource name in each host of an HSMplex. For example, if a 2-host HSMplex and a 3-host HSMplex share a GRS ring, then apply the same patches to both systems in the 2-host HSMplex or to all three systems in the 3-host HSMplex. One HSMplex can use the DFSMShsm-provided names.

For more information about sysplex environments, see Chapter 12, “DFSMShsm in a sysplex environment,” on page 279.

Patching to force UCBs to be OBTAINed each time a volume is space checked

This patch causes the DFSMShsm space checking function to re-OBTAIN each DASD volume's UCB prior to addressing fields within the structure. This prevents errors in an environment where dynamic I/O configurations are occurring and the user does not want to stop and start DFSMShsm.

Issue the following PATCH command to force the UCBs to be OBTAINed for each DASD volume processed during space checking:

```
PATCH .MCVT.+4F1 BITS(......1.)
```

Once it is known that no further dynamic I/O configurations will be occurring, this bit should be reset to OFF. Leaving this bit set on will impact the performance of your system. Issue the following PATCH command to set the bit OFF:

```
PATCH .MCVT.+4F1 BITS(......0.)
```

Running conditional tracing

Conditional tracing allows users to turn off some of the problem determination aid (PDA) tracing and CAPACITYMODE tracing that DFSMShsm normally performs.

If conditional tracing is OFF, performance improves. If conditional tracing is ON, serviceability improves. Users must be aware, however, that if any tracing is off, then data capture on the first failure may be compromised, and may require a problem recreation.

The default for tracing is to trace everything.

Three tracing functions can be turned off with the following PATCH commands:

```
 PATCH .MCVT.+558 BITS(0.......) /* OFF=Do not trace CPOOL calls   */
                                 /* (GETCELL or FREECELL)          */


PATCH .MCVT.+558 BITS(.0......)  /* OFF=Do not trace entries with         */
                                 /* CONDitional or CAPACITYMODE specified */


PATCH .MCVT.+558 BITS(..0.....)  /* OFF=Do not trace entries of       */
                                 /* REJECTION during volume selection  */
```

Using the tape span size value regardless of data set size

If you prefer to have a portion of unused tape remain at the end of your backup tapes rather than having data sets span tapes, DFSMShsm has a patchable bit in the MCVT that allows you to do this.

When this bit is set ON, DFSMShsm uses the SETSYS TAPESPANSIZE value (a number between 0 and 4000 in units of megabytes) to either start the data set on a new tape or to start the data set on the current tape and allow it to span more tapes. However, DFSMShsm does not consider tape capacity. DFSMShsm does consider the capacity of the target output device before it checks the SETSYS TAPESPANSIZE value.

To request DFSMShsm to use the tape span size value, issue the following PATCH command:

```
PATCH .MCVT.+4F1 BITS(.......1) /* ON = Use the tape span size value  */
                                /* regardless of the data set size    */
```

For more information about the SETSYS TAPESPANSIZE command, see the DFSMShsm topic in *z/OS DFSMSdfp Storage Administration*.

Updating MC1 free space information for ML1 volumes after an return code 37 in a multi-host environment

If the selected ML1 volume fails with return code 37 and this is the last try, the failing volume is LSPACed and the MC1 free space is updated if the volume is active. The remaining ML1 volumes are LSPACed. The MC1 is updated for those volumes in which the MVT free space is below the number of tracks specified in the MCVT_L1FRESP field and the free space has changed by more then the number of tracks specified in the MCVT_L1SD field.

Patch the MCVT_L1FRESP higher or lower depending on the free space level at which you want to trigger an immediate update of the MC1 for active volumes.

Guideline: Set the MCVT_L1FRESP field to a value at least two times the size of a backup VTOC copy data set. This helps prevent out-of-space conditions during volume backup, volume dump and FREEVOL. For example:

```
PATCH .MCVT.+560 X'00001388'
```

Note: MCVT_L1FRESP is initialized to 4500 tracks at startup.

If the same volume continues to be selected when there are other volumes with more free space, patch MCVT_L1SD lower. For example:

```
PATCH .MCVT.+564 X'000000C8'
```

Note: Note: MCVT_L1SD is initialized to 500 tracks at startup.

Allowing DFSMShsm to use the 3590-1 generic unit when it contains mixed track technology drives

You may have conditions where you need to use the 3590-1 generic unit but it contains a mixture of 3590 devices that cannot share tapes. When this condition occurs, you must use other means, such as SMS ACS routines, to keep these drives separate and you can use the following patch to disable the DFSMShsm compatibility checking:

```
PATCH .MCVT.+3D5 BITS(.1......)
```

You can use the following patch to re-enable the DFSMShsm compatibility checking:

```
PATCH .MCVT.+3D5 BITS(.0......)
```

By default, the checking is done for non SMS tape allocations. When the checking is enabled, non SMS tape allocations for generic unit 3590-1 containing mixed track technologies cause ARC0030I to be issued. The allocation is allowed to proceed when the message is issued, but results in an OPEN failure if a tape/tape unit mismatch occurs. ARC0030I is not issued for a mix of tape units sharing a common write format, for example 3592-2 with 3592-3E.

Allowing functions to release ARCENQG and ARCCAT or ARCGPA and ARCCAT for CDS backup to continue

Certain functions and modules enqueue ARCENQG and ARCCAT or ARCGPA and ARCCAT resources during processing. When these resources are held, CDS backup cannot continue. Table 50 lists the external interface (patch) that can be enabled or tuned to set an interval at which the function or module is forced to release the resources.

When a patch is used, the function or module is quiesced and the resource is released. After the set interval, a hold on the resource is obtained again. These patches are not applicable for every action of the function or module. For example, the auditing 500 data set interval is not applicable for audit functions that do not process data sets.

Table 50. Functions and the Patch Used to Control the Release Interval to Allow CDS Backup to Continue

Function or module	Resource released	Default interval	External interface (patch)
Recycle	ARCGPA, ARCCAT	15 minutes	`PATCH.YGCB.+CE X'`*nn*`'`
Audit	ARCGPA, ARCCAT	500 data sets	`PATCH.MCVT.+25C X'`*nn*`'`
TAPECOPY	ARCGPA, ARCCAT	10,000 blocks	`PATCH.MCVT.+286 BITS(........1.)`
Migration (secondary space management)	ARCGPA, ARCCAT	5 minutes (TIME default)	`PATCH.MGCB.+B2 X'`*nn*`'`
Migration (primary space management, interval migration, on-demand migration, and command migration)	ARCENQG, ARCCAT	0 minutes (TIME default)	`PATCH.MGCB.+90 X'`*nn*`'`
EXPIREBV	ARCGPA, ARCCAT	500 data sets	`PATCH.MCVT.+25C X'`*nn*`'`
Tape device allocation (migration)	ARCGPA, ARCCAT	10 seconds	`PATCH.MCVT.+194 BIT(.......1)`

Table 50. Functions and the Patch Used to Control the Release Interval to Allow CDS Backup to Continue (continued)

Function or module	Resource released	Default interval	External interface (patch)
Note: 1. These patches to not have an effect on releasing the ARCCAT resource under normal operation of the function or module. However, if an error occurs that prevents the function or module from releasing ARCCAT when a CDS backup is initiated, the patch value will apply. 2. X'nn' is the hexadecimal representation of the interval (minutes or data sets) for which the resource is released. 3. For both migration patches (in both a non-RLS and RLS environment), when work is completed on the current data set, a check is performed to determine whether the number of minutes specified in the patch has passed. If the number of minutes has passed, the resource is released so that other functions (such as CDS backup) that need the resource are able to continue. 4. For both migration patches (in an RLS environment), when work is completed on the current data set, an additional check is performed to determine whether any other host is waiting for the resource to be available before starting a CDS backup. If a host is waiting for the resource, the resource is released.			

Suppressing SYNCDEV for alternate tapes during duplex migration

You can suppress SYNCDEV processing for alternate tapes during duplex migration. To do so, enter the following PATCH command:

```
PATCH .MCVT.+196 BITS(..1.....)
```

To reactivate SYNCDEV processing for alternate tapes, enter the following command:

```
PATCH .MCVT.+196 BITS(..0.....)
```

By default, SYNCDEV processing occurs for alternate tapes.

This patch is not suggested for earlier technology tape drives, such as the 3490 and 3590. However, using it with the 3592-J or 3592 tape drives can improve performance.

This patch is ignored if your installation has specified that both the original and alternate tapes are to be marked full if the alternate tape volume encounters an error. You can specify this option through the ERRORALTERNATE(MARKFULL) subparameter of the SETSYS DUPLEX MIGRATION(Y) command.

Patching to allow building of a dummy MCD record for large data sets whose estimated compacted size exceeds the 64 KB track DASD limit

Issue the following patch if you want DFSMShsm to build a dummy MCD record for a large data set when the initial migration attempt fails with an ABENDSB37 because the data set did not compact well:

```
PATCH .MCVT.+595 BITS(1.......)
```

The dummy MCD record is used to determine if subsequent attempts to migrate the data set should be allowed. If the estimated compacted size, based on the current size and saved compaction history, exceeds the 64 KB track limit, the migration will fail with the following messages: ARC0734I RC=8, REASON=10, or with ARC1001I RC=0008, REAS=0010 and ARC1208I. The RC0008 and REAS0010 indicate that the DFSMShsm-owned copy of the data set to be migrated is estimated to be greater than 64K tracks. The data set is eventually migrated to ML2 tape when eligible.

Note: The dummy MCD record is built for data sets with an original size greater than 64K tracks that don't compact well when migrating. It is also built for data sets whose size is less than 64K tracks but may grow beyond the 64 KB track DASD limit on the migration volume due to reblocking and control data.

Allowing DFSMShsm to honor an explicit expiration date even if the current management class retention limit equals 0

The installation has the option of allowing DFSMShsm to expire data sets based on an explicit expiration date found to be invalid because the current Management Class definition does not permit explicit expiration dates. You can use the following PATCH command to request that DFSMShsm honor an explicit expiration date, even if the current Management Class definition specifies a RETENTION LIMIT value of zero. By default, this type of invalid expiration date is ignored.

```
PATCH .MCVT.+1B5 BITS(.....1..)
```

Using the generic rather than the esoteric unit name for duplex generated tape copies

DFSMShsm will pass the appropriate SETSYS specified unit name, for example SETSYS MIGUNITNAME (esoteric or generic unit name) in effect when a duplex failure generates a tape copy, instead of the generic equivalent of the SETSYS unit name, for example 3590-1. This change affects DFSMShsm's Backup, Migration and Recycle functions. The ABARS function is not affected.

To change the behavior back to always passing the generic unit for the tape copy, use the DFSMShsm PATCH command to set the MCVTF_GENERIC_TCN_UNIT flag on:

```
PATCH .MCVT.+196 BITS(....1...)
```

To return to the default behavior of using the SETSYS specified unit name:

```
PATCH .MCVT.+196 BITS(....0...)
```

Modifying the allocation quantities for catalog information data sets

Catalog information data sets (CIDS) are created on ML1 volumes during FRBACKUP of a copy pool that is defined to capture catalog information. By default, DFSMShsm will attempt to allocate a CIDS with 50 primary and 50 secondary cylinders. Factors such as limited space on ML1 volumes or a higher number of catalog entries in the catalogs to be captured may require changes to these values.

The number of cylinders specified for primary allocation of the CIDS can be adjusted with a patch to the FRGCB. For example:

```
PATCH .FRGCB.+30C X'00000064' VERIFY(.FRGCB.+30C X'00000032')
```

The number of cylinders specified for secondary allocation of the CIDS can be adjusted with a patch to the FRGCB. For example:

```
PATCH .FRGCB.+310 X'00000064' VERIFY(.FRGCB.+310 X'00000032')
```

Enabling volume backup to process data sets with names ending with .LIST, .OUTLIST or .LINKLIST

When volume backup processing backs up data sets for a specific volume, DFSMShsm skips data sets with names that end with:

.LIST
.OUTLIST
.LINKLIST

To request that these data sets not be skipped, issue the following PATCH command:

```
PATCH .MCVT.+297 BITS('1.......')
```

Prompting before removing volumes in an HSMplex environment

When removing a volume from the control of DFSMShsm in an HSMplex environment, a DELVOL command must be issued on each DFSMShsm host in the HSMplex. If a DELVOL command is not issued on a host, the volume might continue to be managed by that host even though the other hosts are no longer managing the volume.

To help ensure that the volume is removed from the control of each DFSMShsm host, the MCVTF_DELVOL_WTOR bit can be turned on to cause the WTOR message ARC0265I to be issued for each DELVOL command. Message ARC0265I is a reminder that the DELVOL command must be issued on each DFSMShsm host in the HSMplex and requests confirmation from the user to process the DELVOL command. If confirmed, DFSMShsm will process the DELVOL command on the corresponding host. Otherwise, the DELVOL command is not processed. It is the responsibility of the user to issue the DELVOL command on each host. If this patch is not enabled, WTOR message ARC0265I is not issued before processing the DELVOL command.

To enable prompting before removing volumes in an HSMplex environment, issue the following PATCH command:

```
PATCH .MCVT.+297 BITS(...1....)
```

Returning to the previous method of serializing on a GDS data set during migration

DFSMShsm uses a method for serializing the migration of a GDS data set that does not require locking the GDG base. This method overrides patches to the MCVT at offset X'4C3', bit 7 (......X.). If you want DFSMShsm to return to the previous serialization method, issue the following patch command:

```
PATCH .MCVT.+595 BITS(.....0..)
```

When this bit is off (0), the GDG base is locked if the patchable bit in the MCVT at offset X'4C3', bit 7 (......X.) is off (0). The base is not locked if the bit is on (1). If disabled, the new serialization method can be re-enabled with the following patch:

```
PATCH .MCVT.+595 BITS(.....1..)
```

Allowing DFSMShsm to create backup copies older than the latest retained-days copy

DFSMShsm allows the creation of a backup copy, using the BACKDS command with NEWNAME parameter, that has a date and time older than the latest backup-retained-days copy. You must set an enabling bit, as follows:

```
PATCH .ARCCVT.+5D4 BITS(...1....)
```

You disable the function as follows:

```
PATCH .ARCCVT.+5D4 BITS(...0....)
```

Once the patch is set (enabled), you can issue a BACKDS command with NEWNAME and RETAINDAYS parameters that specify a date and time that is older than the most current retained-days copy. The syntax is as follows:

`BACKDS` *dsname* `NEWNAME(`*newdsname*`) DATE(`*yyyy/mm/dd*`) TIME(`*hhmmss*`) RETAINDAYS(`*days*`)`

Only a single MCBR record is supported. A single MCBR record can hold 100 backup versions. Once DFSMShsm fills a single MCBR record, it will again fail additional copies that come in out of chronological order (new versions with a backup date older than the most recent version in the MCBR).

The total number of backup versions that can be added out of chronological order is 100+*nn*, where *nn* is the value for VERSIONS set with the SETSYS command. For example, issuing SETSYS VERSIONS(100) allows the creation of 200 versions out of chronological order.

This feature is available with the BACKDS command only when NEWNAME, DATE, TIME, and RETAINDAYS are specified. For more information, refer to BACKDS command in *z/OS DFSMShsm Storage Administration*.

Enabling or disabling RC 20 through RC 40 ARCMDEXT return code for transitions

The MGCBF_MDEXT control flag is used to manage RC=20 through RC=40 ARCMDEXT return code for class transitions. If MGCBF_MDEXT flag is set ON, and ARCMDEXT exit returns RC=20 through RC=40 return codes, the class transition function is converted to the migration function. In this case, the data set is migrated according to the ARCMDEXT return code. To set MGCBF_MDEXT control flag ON, use the he following command:

```
PATCH .MGCB.+111 BITS(.....1..)
```

To set MGCBF_MDEXT control flag OFF, use the following command:

```
PATCH .MGCB.+111 BITS(.....0..)
```

Enabling FSR records to be recorded for errors, reported by message ARC0734I, found during SMS data set eligibility checking for primary space management

To specify that FSR records are to be recorded for errors, reported by message ARC0734I, found during SMS data set eligibility checking for primary space management, issue the following PATCH command:

```
PATCH .MGCB.+EF BITS(......1.)
```

To specify that the FSR records are not to be recorded, issue the following PATCH command:

```
PATCH .MGCB.+EF BITS(......0.)
```

By default, the FSR records are not recorded when an ARC0734I message is reported during SMS data set eligibility checking for primary space management.

Chapter 17. Special considerations

This topic contains the following information that you want to consider before you install the DFSMShsm program:

For information on additional special considerations, refer to *z/OS DFSMShsm Storage Administration*.

Backup profiles and the RACF data set

When you choose the SETSYS PROFILEBACKUP command, DFSMShsm creates backup profiles of RACF discrete data set profiles when the corresponding data sets are backed up. These backup profiles have the same data set name as the backup version. Therefore, all of the backup profiles have the same high-level qualifier for their data set names—the DFSMShsm backup prefix. Your installation can define the prefix or you can use the DFSMShsm default of HSM.

When RACF stores a data set profile in its data set, the location of the profile in the data set is based on the data set name. If you do not take any action to avoid storing the backup profiles in the same general location in the RACF data set, the overall system performance can be degraded. Whether you should take action or not depends on how many RACF-protected discrete data sets DFSMShsm backs up and what release of RACF you have installed on your system.

Increasing VTOC size for large capacity devices

When selecting large capacity devices (for example, 3380 and 3390), be aware that the number of one-track data sets will probably exceed that of a TSO volume. Therefore, ensure that the VTOC is increased to accommodate the many data sets on these devices.

DFSMShsm command processor performance considerations

The DFSMShsm command processor task attaches separate tasks to process the following long-running commands:

- AUDIT
- BACKVOL CDS

- LIST
- RECYCLE
- REPORT
- TAPECOPY
- TAPEREPL
- EXPIREBV

Other commands are processed by using calls to the appropriate DFSMShsm modules. If additional commands are received by the general command processor before control has been returned to it by one of the above command processors, processing of the new command does not begin until control has been returned. The delay in the processing is normally not noticeable. However, if the command currently in process is waiting for a resource held by a long-running task, some processing delay can be experienced. The user should consider this aspect when using long running commands.

Incompatibilities caused by DFSMShsm

Although installing DFSMShsm should not affect your data sets, the following items can cause incompatibilities when DFSMShsm is installed in a system:

- Volume serial number of MIGRAT or PRIVAT
- IEHMOVE utility
- VSAM migration
- RACF volume authority checking
- Licensed programs that allocate existing data sets by specifying the *volser* and *unit* and do not open the data set
- RACF program resource profile
- Processing while DFSMShsm is inactive

Volume serial number of MIGRAT or PRIVAT

DFSMShsm uses the volume serial number of MIGRAT for data sets in the computing system catalog to identify migrated data sets. Therefore, you must not define any volume with a volume serial number of MIGRAT. If you use MIGRAT as a volume serial number, data access failures on the volume can result.

When DFSMShsm needs a scratch tape, the operator receives a mount message for a volume serial number of PRIVAT, which is the standard name for a private scratch tape volume. Therefore, you should not define any tape volumes with a volume serial number of PRIVAT. If you use PRIVAT as a volume serial number, the volume might never be mounted and if it is, data access failures can result.

IEHMOVE utility

An incompatibility can exist between the IEHMOVE utility and DFSMShsm if the data set being moved or copied has migrated.

If the IEHMOVE utility assumes that the data set being copied or moved is cataloged, the volume serial number returned by the catalog locate for the data set must be associated with a volume allocated to the job step, or the IEHMOVE utility cannot complete the request. The IEHMOVE utility assumes that the data set is cataloged if the FROM=*device*=*list* parameter has not been specified. Also, in this case, the located volume serial number is MIGRAT. Therefore, you have to (1) use the RECALL command to recall the migrated data set, or (2) automatically recall the data set by allocating it in a previous step before running the IEHMOVE utility.

TSO ALTER command and access method services ALTER command

An incompatibility can exist between DFSMShsm and the TSO ALTER command or the access method services ALTER RENAME command if the data set being renamed has migrated.

If you do not provide a FILE DD statement for the data set being renamed or you provide a FILE DD statement and do not specify the data set name, DFSMShsm successfully recalls and renames the data set. If you provide a FILE DD statement to allocate the volume and supply the data set name on this DD statement, a message is issued specifying that dynamic allocation has detected a discrepancy between the volume specified with the FILE DD statement and the volume recorded in the computing system catalog, which in this case is MIGRAT. The data set remains migrated and is not renamed.

TSO DELETE command and access method services DELETE command

An incompatibility can exist between DFSMShsm and the TSO DELETE command or the access method services DELETE SCRATCH command if the data set being renamed has migrated.

If you do not provide a FILE DD statement for the data set being deleted or you provide a FILE DD statement and do not specify the data set name, DFSMShsm successfully recalls and deletes the data set. If you provide a FILE DD statement to allocate the volume and supply the data set name on this DD statement, a message is issued specifying that dynamic allocation has detected a discrepancy between the volume specified with the FILE DD statement and the volume recorded in the computing system catalog, which in this case is MIGRAT. If the volume specified on the FILE DD statement is the same one from which the data set last migrated, the data set is scratched. If the volume is not the same, the data set is not scratched. In either case, the data set remains cataloged.

Data set VTOC entry date-last-referenced field

Do not put DFSMShsm-managed data sets on volumes allocated to devices that have the write-inhibit switch on. If the write-inhibit switch is on, the open or end-of-volume routines fail with a file-protect I/O error during the attempt to update the date-last-referenced field. This causes the job to fail and the operator receives an error message.

VSAM migration (non-SMS)

After a VSAM data set has migrated and been recalled, DFSMShsm resets only the date-last-referenced field in the DSCB of the primary volume and only for the data object of the base cluster. The other information about the data set is in the catalog rather than the data set VTOC entry. For example, creation and expiration dates of the base data object are in the catalog.

RACF volume authority checking

An incompatibility can exist between DFSMShsm and RACF if the data set being referred to is migrated. A user with volume authority (operations authority or

DASDVOL authority) for a specific volume can normally alter or delete any data set on that volume. If DFSMShsm is installed and the data set in question is migrated, the operation can fail. The user must also have data set authority for the data set that is being accessed because DFSMShsm checks the user's authority at the data set level.

Accessing data without allocation or OPEN (non-SMS)

If DFSMShsm migrates a data set, subsequent access to the data set while it is migrated must cause allocation to do a catalog LOCATE or OPEN must be issued to cause the data set to be automatically recalled. For example, if the *volser* and *unit* are specified on a SYSPROC DD statement in a TSO logon procedure, a problem exists if that data set is migrated. The problem exists because no catalog LOCATE is done by allocation and no OPEN is performed in TSO unless the volume specified is an SMS-managed volume.

DFSMShsm does not migrate data sets having SYS1 as the first qualifier unless a SETMIG LEVEL request has been issued to remove this restriction. For other data sets used in the above manner, the system programmer must either not specify the *volser* and *unit* (which causes a catalog LOCATE by allocation), or specify the names of the data sets on SETMIG commands placed in the DFSMShsm startup member to prevent them from migrating.

RACF program resource profile

The RACF program resource profile is not supported by DFSMShsm. Migration and subsequent recall of data sets that are protected by this method cause the RACF protection to be invalid because the program resource profile is in a non-updated status.

Update password for integrated catalog facility user catalogs

DFSMShsm is unable to supply the update password for ALTER requests directed at entries in ICF user catalogs. On systems without MVS/DFP 3.1.0 or a subsequent release installed, the system operator is prompted for the password when DFSMShsm needs to update an entry. If the operator supplies an incorrect password, the current DFSMShsm operation fails.

On systems with MVS/DFP 3.1.0 or a subsequent release installed, the update operation occurs without operator intervention.

Processing while DFSMShsm is inactive

During LOCATE, SCRATCH, RENAME, and OPEN processing, certain conditions can be encountered that require intervention by DFSMShsm. Any data set function that may require DFSMShsm participation results in a system request being sent to DFSMShsm. Of course, DFSMShsm user commands entered from TSO are sent to DFSMShsm for processing. If DFSMShsm has been active during the current IPL of the system but is not currently active when the preceding conditions occur, message ARC0050A, ARC0052A, or ARC0055A can be issued to indicate that DFSMShsm must be started to process the system request or user command.

When DFSMShsm is inactive, the messages just listed are *always* issued if the system request detects that a data set is cataloged on volume serial MIGRAT or if a DFSMShsm user command is submitted from TSO. If the system request results in a DSCB-not-found condition, the messages discussed earlier are issued *only* if DFSMShsm is *not* in debug mode at the time of DFSMShsm shutdown.

DFSMShsm abnormal end considerations

The following are some suggested actions to take when DFSMShsm ends abnormally.

Recovering DFSMShsm after an abnormal end

When an error causes DFSMShsm to end abnormally, SETSYS SYS1DUMP is the DFSMShsm default and causes a dump to be written to a SYS1.DUMP data set. SYSUDUMP, SYSMDUMP, or SYSABEND DD are other options, but their use is strongly discouraged. You can analyze the dump to determine how much of the current function has been processed and how much work is waiting to be processed. You might need to reschedule the function that was running when the abnormal end occurred. If you do not include a SYSUDUMP, SYSMDUMP, or SYSABEND DD statement, print and edit the DFSMShsm log to determine and re-create the events that caused the abnormal end. You can use the TRAP command to isolate the situation that has caused the abnormal end.

Recovering from an abnormal end of a DFSMShsm subtask

The data set currently being processed fails and processing continues with the next eligible data set on the volume. However, if the ARCCP subtask were to abnormally end, DFSMShsm might have lost some control information that must be reentered. The type of information lost to DFSMShsm is that given to DFSMShsm after it is initialized. This information includes:

- All primary and migration level 1 volumes added to DFSMShsm control that are not in the ARCCMD*xx* parameter library member
- Any change to the space management status of a data set, a group, or all data sets so automatic or command space management does not occur for those data sets

Other changes made to DFSMShsm since it was initialized are not lost when DFSMShsm abnormally ends.

Recovering from an abnormal end of the DFSMShsm main task

If the DFSMShsm ESTAE retry routine cannot recover from the error, DFSMShsm stops. You must restart DFSMShsm to continue processing.

Restarting DFSMShsm after an abnormal end

You can use the RESTART keyword in the DFSMShsm startup procedure to automatically restart DFSMShsm. For more information about automatically restarting DFSMShsm after an abnormal end, see "Using the RESTART keyword to automatically restart DFSMShsm after an abnormal end" on page 306.

After DFSMShsm is restarted, VSAM error messages are issued when DFSMShsm attempts to open the control data sets, an access method services VERIFY command is issued, and the open is retried. Using the VERIFY command is usually successful, and DFSMShsm continues processing normally from this point. If DFSMShsm cannot initialize successfully or if the operator cannot restart DFSMShsm, this probably means that one of the control data sets has been damaged during the abnormal end. After you analyze the dump resulting from the abnormal end and you determine which control data set is damaged, you can recover it. To recover the control data set, issue the access method services IMPORT command to make the most recent backup copy available. Then, start DFSMShsm and issue the UPDATEC command to combine the latest transactions in the journal data set with the restored backup copy of the control data set.

If the journal data set is damaged, you must stop DFSMShsm, reallocate a new journal data set, restart DFSMShsm, and back up the control data sets.

If the journal data set and the control data sets are damaged, you might need to use the AUDIT or FIXCDS commands to make your records correct in the control data sets.

For more information about recovering control data sets, see *z/OS DFSMShsm Storage Administration*.

Suppressing duplicate dumps

DFSMShsm uses the dump analysis and elimination (DAE) function to suppress duplicate dumps. To suppress duplicate dumps, issue the SETSYS SYS1DUMP command and then ensure that the SYS1.PARMLIB member ADYSET*xx* is coded with the keyword SUPPRESSALL. For example:

```
DAE=START,RECORDS(400),SVCDUMP(MATCH,UPDATES,SUPPRESSALL)
```

DAE with sysplex scope allows a single DAE data set to be shared across systems in a sysplex. Coupling services of the cross-system coupling facility and global resource serialization must be enabled in order for the DAE data set to be shared and dumps to be suppressed across MVS hosts using DFSMShsm. For information about setting up the shared DAE data set, refer to *z/OS MVS Diagnosis: Tools and Service Aids*.

Note: Only those hosts with DFSMShsm release 1.4.0 or greater can use DAE suppression for DFSMShsm dumps.

Duplicate data set names

In an SMS environment, duplicate data set names are not allowed. Therefore, the following description applies only to a non-SMS environment.

Because duplicate data set names can cause confusion and possibly cause the wrong data set to be accessed, each data set, cataloged or uncataloged, should have a unique name. If you recover a cataloged data set but inadvertently include the RECOVER FROMVOLUME command and DFSMShsm backed up an uncataloged data set with the same name, DFSMShsm recovers the uncataloged version.

Another problem can occur with duplicate data set names when you access an uncataloged data set on multiple volumes and a migrated, cataloged, data set exists with the same name. In the JCL, you list the volume serial numbers of the volumes where that data set resides. If, however, the data set does not reside on one of those volumes, a DSCB-not-found condition occurs. DFSMShsm is then asked to determine if that data set name is migrated and to recall the data set if it is migrated. Because DFSMShsm does not know that this is in the middle of an uncataloged data set process, it recalls the cataloged, migrated data set and passes control back to open processing. The results of this are indeterminable.

Debug mode of operation for gradual conversion to DFSMShsm

You can use the debug mode of operation to monitor the effect of DFSMShsm on your computing system. You specify the DEBUG parameter of the SETSYS command to prevent data set movement or deletion from occurring during volume processing.

In debug mode, DFSMShsm simulates volume processing without moving data or updating the control data sets. Messages about volume processing are printed in the activity log and, optionally, at the console. This allows you to monitor the data sets and volumes that are managed if you have specified the NODEBUG parameter of the SETSYS command. As a result of these simulated volume processes, you can determine which data sets you want to prevent DFSMShsm from processing.

To use the debug mode, decide which volumes you want DFSMShsm to manage. First, identify which volumes in your computing system qualify as level 0 volumes. Candidates for level 0 volumes are those volumes containing data sets that should be processed by space management, backup, and dump. Then, based on the number of level 0 volumes that you want DFSMShsm to manage and the amount of data set activity you anticipate on the level 0 volumes, you can decide how many migration level 1, migration level 2, daily backup, spill backup, and dump volumes your computing system requires. After you add these volumes to DFSMShsm, regularly monitor the use of space on all the volumes managed by DFSMShsm so you can add more volumes as your computing system needs them.

After you decide which volumes DFSMShsm is to manage, run DFSMShsm in debug mode. When you are satisfied that the data sets and volumes are being managed as you would like them to be, specify the NODEBUG parameter of the SETSYS command to enable DFSMShsm to actually begin processing the data sets and volumes. You can use this gradual conversion procedure whenever you put additional volumes under DFSMShsm control.

If you issue ABACKUP or ARECOVER commands while DFSMShsm is in debug mode, the commands operate as though you have used the VERIFY parameter.

Data sets that are indicated as migrating or backed up in debug mode can be recalled and recovered with a message that the recall or recovery occurred.

Generation data groups

The following information applies to generation data group management, and specifically to generation data group deletion.

Handling of generation data sets

If each generation of a non-SMS-managed generation data group is on a volume managed by DFSMShsm, each generation is, with some exceptions, managed the same as any other data set.

Note: SMS-managed data sets are managed according to the management class to which they are assigned.

You cannot specify a relative generation data group generation (+ 1) on a command. You can only specify the full generation data group generation (for example, G0011V01).

Because all generation data group data sets have G*nnnn* V*nn* as their last qualifiers, the next to the last qualifier is an indicator of the type of data set being processed. Therefore, when DFSMShsm searches the compaction names tables to determine which encode or decode tables to use, DFSMShsm uses the next to the last qualifier of the data set name to search the compaction names table if the data set is a generation data group data set.

Each generation of a generation data group appears to DFSMShsm as a unique data set. There is no correlation from one generation of a generation data group to another. As a result, you might want to control the number of generations to be kept for a user. For example, if a user is allowed five generations and you have defined five backup versions with the VERSIONS parameter of the SETSYS or ALTERDS commands, the user can have as many as 25 (five versions multiplied by five generations) backup versions of the same generation data group if each generation has been updated five times.

If a generation is created that exceeds the limit of generations to be kept, the oldest generation is deleted at the end of the step or job, unless the oldest generation is protected by:

- A password
- An expiration date that has not yet passed
- RACF (when the user is not authorized to scratch the generation)

In these cases, the oldest generation is not scratched.

If the oldest generation is deleted or uncataloged, the delete operation scratches the oldest generation without recalling it.

Access method services DELETE GDG FORCE command

Assume that you have a generation data group with generation data sets that have migrated, and you wish to delete the group. If you use the Access Method Services DELETE GDG FORCE command to delete the catalog entries for the generation data group and its generation data sets, IDCAMS does not invoke DFSMShsm. You must then issue the DFSMShsm DELETE command for each of the now-uncataloged migration copies.

ISPF validation

If you use ISPF with DFSMShsm, be aware that ISPF validates a data set as being available before an OPEN request is processed. ISPF does not request a recall for the migrated data set name found in the VOL=SER=*dsname* parameter of the DD statement unless the specified volume is an SMS-managed volume. Make sure the migrated data set is recalled before processing the ISPF procedures.

Preventing migration of data sets required for long-running jobs

If you have a multiple-processor system but do not communicate data set allocation to all DFSMShsm systems, you should prevent the migration of data sets that must be available for jobs like CICS® and JES3. Because these jobs are very long-running (they can run for days), the data sets may be open long enough to become eligible for migration. For information on how to prevent the migration of data sets, refer to *z/OS DFSMShsm Storage Administration* under the heading “Controlling Migration of Data Sets and Volumes” for non-SMS-managed data sets, or under the heading “Specifying Migration Attributes” for SMS-managed data sets.

SMF considerations

Normally, System Management Facilities (SMF) records are written to the SMF data sets at DFSMShsm step termination. Because DFSMShsm uses dynamic allocation and processes thousands of different data sets each day, this process can take a very long time. Also, these records are kept in DFSMShsm address space which can lead to the exhaustion of virtual storage.

For these reasons, use the SMF option to avoid doing DD consolidation. If you do use DD consolidation, request that it be done periodically by requesting SMF interval recording for started tasks. Refer to *z/OS MVS System Management Facilities (SMF)* for an explanation of how to request these options.

DFSMSdss address spaces started by DFSMShsm

In order to maximize throughput, DFSMSdss address spaces are started by DFSMShsm for certain functions. Table 51 defines the DFSMSdss address space identifiers for address spaces started by DFSMShsm.

DFSMSdss requires that the ARC* and DFSSFR* address spaces run at the same or higher priority as DFSMShsm. You can create WLM profiles so that these address spaces are assigned to the proper service classes

Table 51. DFSMSdss address space identifiers for address spaces started by DFSMShsm functions

Function	Address space identifier
Backup	ARC*n*BKUP
CDS backup	ARC*n*CDSB
Dump	ARC*n*DUMP
Fast replication backup	DSSFRB*xx*
Fast replication recovery (copy pool and volume)	DSSFRR*xx*
Fast replication recovery (data set)	DSSFRDSR
Migration	ARC*n*MIGR
Recovery from backup (data set and full-volume)	ARC*n*RCVR
Recovery from a dump tape (data set)	ARC*n*REST
Recovery from a dump tape (full-volume)	ARC*n*RST*y*

Variables used in the address space identifiers are defined as follows:

n is the unique DFSMShsm host ID.

xx is the instance of the DFSMSdss started task for fast replication backup and fast replication recovery. The value is a two-digit number 01 - 64.

y is the instance of the DFSMSdss started task for full-volume recovery from a dump tape. The value is a number 1 - 4.

For example, migration for DFSMShsm host ID 1 results in a generated address space identifier of `ARC1MIGR`.

Note:

1. Use of the DFSMSdss cross memory API for these functions might result in an increase in CPU time.

2. Once DFSMShsm starts one or more of these address spaces, the address spaces remain active until DFSMShsm is terminated. When DFSMShsm terminates, all of the started DFSMSdss address spaces automatically terminate.
3. The FRBACKUP and FRRECOV commands always use DFSMSdss cross memory support when backing up and recovering volumes to and from disk. Note that FRBACKUP processing, automatic dump processing, and FRRECOV processing all invoke DFSMSdss to dump from fast replication volumes to tape, or to restore from tape. DFSMSdss address space identifiers can be started optionally, based on the SETSYS DSSXMMODE command in the ARCCMDxx member of the SYS1.PARMLIB. For more information on using the SETSYS DSSXMMODE command and on parallelism for fast replication, see *z/OS DFSMShsm Storage Administration*.
4. The DFSMSdss address space identifiers started for the dump, data set recovery from dump, data set recovery from backup, migration, backup, and CDS backup functions are optional and are controlled using the SETSYS DSSXMMODE command in the ARCCMDxx member of SYS1.PARMLIB. For more information on using the SETSYS DSSXMMODE command, see *z/OS DFSMShsm Storage Administration*.

Chapter 18. Health Checker for DFSMShsm

The IBM Health Checker for z/OS includes checks for DFSMShsm. These checks are designed to help you determine if DFSMShsm is correctly configured and is consistent with IBM's recommendations.

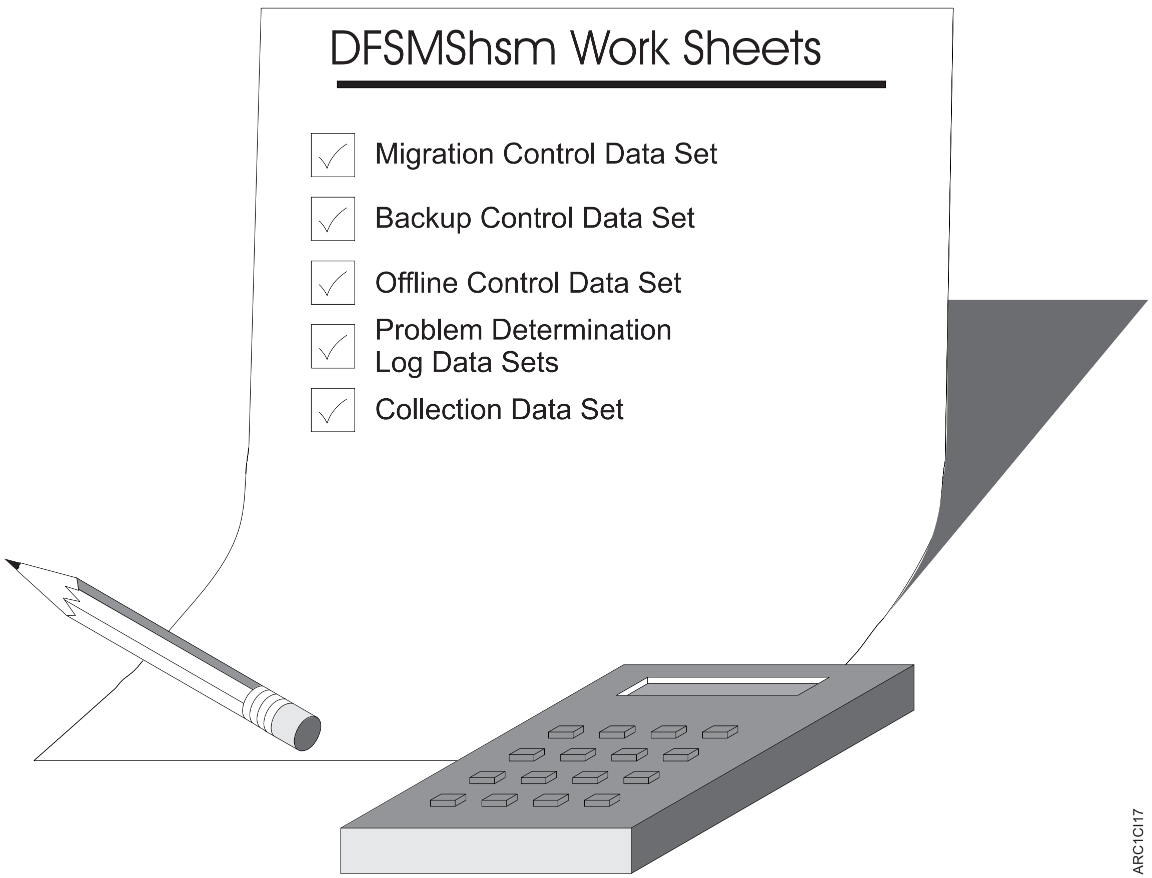
For more information, see *z/OS DFSMShsm Storage Administration* and *IBM Health Checker for z/OS User's Guide*.

Part 3. Appendixes

Appendix A. DFSMShsm work sheets

The following work sheets help you determine the amount of storage that you need.

All examples in this appendix are based on 3390 DASD.



MCDS size work sheet

Use the following work sheet (Figure 99 on page 388) to calculate the size for your MCDS.

Migration Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = *mds* - Number of data sets that you want to migrate.

2. Substitute the value for *mds* in the following calculation. This is the space for your current MCDS records.

   516 x (*mds*= ________) = *subtotal* = ______________

3. Multiply the *subtotal* by 1.5 to allow for additional MCDS growth.

   subtotal x 1.5 = *total* = ______________

Total = **total number of bytes for the MCDS**

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the MCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the MCDS.

$$\frac{\text{Total bytes used by the MCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{\text{(Total = \_\_\_\_\_\_\_\_)}}{737280} = \_\_\_\_\_\_\_\_\_\_\_\_\_\_$$

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the migration control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C156

Figure 99. Migration Control Data Set Size Work Sheet

BCDS size work sheet

Use the following work sheet (Figure 100 on page 389) to calculate the size for your BCDS.

Backup Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = *bver* - Number of backup versions of each data set. This same number is used with the VERSION parameter of the SETSYS command or is specified in management classes. The upper bound for *bver* is either 29 or 100, depending on the maximum record size in the BCDS definition.

   ________ = *nds* - Number of data sets backed up automatically.

2. Substitute the values for *bver* and *nds* in the following calculation. This is the space for your current BCDS records.

   398 x *(bver=* ________ *)* x *(nds=* ________ *) = subtotal* = ______________

3. Multiply the *subtotal* by 1.5 to allow for additional BCDS growth.

   subtotal x 1.5 = *total* = ______________

Total = **total number of bytes for the BCDS**

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the BCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the BCDS.

   $$\frac{\text{Total bytes used by the BCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{\text{(Total = \_\_\_\_\_\_\_\_)}}{737280} = \text{\_\_\_\_\_\_\_\_\_\_\_\_\_\_}$$

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the backup control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C155

Figure 100. Backup Control Data Set Size Work Sheet

OCDS size work sheet

Use the work sheet in Figure 101 on page 390 to calculate the size for your OCDS. The work sheet assumes a larger record size (6144 bytes) to enable 106 data set entries for extended TTOCs. If you do not intend to use extended TTOCs, use 2048 bytes for the record size and 33 for the data set entries.

Offline Control Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   __________ = bver - Number of backup data set versions that the volume contains.

   __________ = mds - Number of migration data set copies the volume contains.

   __________ = nds - Number of data sets backed up automatically.

   __________ = n - Total number of backup version and migration copy data sets for your installation.

2. Substitute the value for *n* in the following calculation. This is the space for your current OCDS records.

   $$\frac{n}{106} \times 6144 = \text{subtotal} \qquad = \_\_\_\_\_\_\_\_\_\_$$

3. Multiply the *subtotal* by 1.5 to allow for additional OCDS growth.

   subtotal x 1.5 = *total* = __________

4. Divide the total number of bytes per cylinder (using 3390 as an example) into the total number of bytes required by the OCDS. If the result is a fraction, round up to the next whole number. This is the number of cylinders you should allocate for the OCDS.

   $$\frac{\text{Total bytes used by the OCDS}}{\text{Total bytes per cylinder (3390)}} = \frac{\text{(Total = \_\_\_\_\_\_\_\_\_\_)}}{737280} = \_\_\_\_\_\_\_\_\_\_$$

Note: 737280 is the total number of bytes for each cylinder of a 3390, assuming FREESPACE (0 0). This value is based on the DATA CONTROLINTERVALSIZE (CISIZE) for the offline control data set shown in the starter set. Because the CISIZE is 12288 (12K), the physical block size is 12KB, which allows 48KB per track or 720KB per cylinder. With no FREESPACE per cylinder, the resulting space for data is 720KB or 737280.

ARC1C157

Figure 101. Offline Control Data Set Size Work Sheet

Problem determination aid log data set size work sheet—Short-term trace history

Use the work sheet in Figure 102 on page 391 to calculate the size of your PDA log data set (short term).

PDA Log Data Set Size Work Sheet
Short-Term Trace History

1. Fill in the following blanks with values for your installation.

________	= ?tracehours	- The number of hours of trace history you want to retain.
________	= ?UID	- The high-level qualifier you want to use for the PDA log data sets.
________	= ?HOSTID	- The identifier for the processing unit at your site.
________	= ?TRACEUNIT	- The unit identifier for the device on which you want to allocate the PDA log data sets.
________	= ?TRACEVOL	- The serial number for the volume on which you want to put your PDA log data sets.

2. Allocate the minimum recommended storage for PDA log data sets: 20 cylinders.

   The following example allocation can be seen in the starter set. Substitute the values you used in step 1 of this work sheet, and run the following JCL job to allocate and catalog the PDA log data sets.

```
//ALLOPDO   JOB MSGLEVEL=1, TYPRUN=HOLD
//STEP1     EXEC PGM=IEFBR14
//DD1       DD DSN=&UID..H?HOSTID..HSMPDOX,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//             VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
//DD2       DD DSN=&UID..H?HOSTID..HSMPDOY,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//             VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
```

   If you have allocated these data sets as SMS-managed, they must be allocated on a specific volume and they must be associated with a storage class having the GUARANTEED SPACE attribute.

3. Measure the cylinders per hour trace history generation rate at your site.

   After one hour of processing (during a time of high DFSMShsm activity), measure the amount of storage used to record that hour's trace activity. Issue the following SWAPLOG command to swap the ARCPDOX and ARCPDOY data sets. After you have swapped these data sets, the ARCPDOY data set will be ready to measure and the ARCPDOX data set will be ready to receive additional trace data.

```
SWAPLOG PDA
```

Cylinders/hr = **cylinders per hour of trace history**

4. Calculate the total amount of cylinders required for your site's trace history data.

   ((tracehours = ________) x (cylinders/hr= ________)) = ________

Total = **total number of cylinders of trace data**

5. Divide in half the total cylinders required for your short-term trace history interval. If the result is a fraction, round up to the next whole number.

   $\frac{(Total = \qquad)}{2}$ = ________

Total number of cylinders to allocate for each data set

ARC1C158

Figure 102. Problem Determination Aid Log Data Set Size Work Sheet—Short-Term Trace History

Problem determination aid log data set size work sheet—Long-term trace history

Use the following work sheet (Figure 103) to calculate the size of your PDA log data set (long term).

PDA Log Data Set Size Work Sheet
Long-Term Trace History

1. Fill in the following blanks with values for your installation.

   ________ = ?UID - The high-level qualifier you want to use for the PDA log data sets.

   ________ = ?HOSTID - The identifier for the processing unit at your site.

   ________ = ?TRACEUNIT - The unit identifier for the device on which you want to allocate the PDA log data sets.

   ________ = ?TRACEVOL - The serial number for the volume on which you want to put your PDA log data sets.

2. Allocate the minimum recommended storage for PDA log data sets: 20 cylinders.

   The following example allocation can be found in the starter set. Substitute the values you used in step 1 of this work sheet, and run the following JCL job to allocate and catalog the PDA log data sets.

```
//ALLOPDO   JOB MSGLEVEL=1, TYPRUN=HOLD
//STEP1     EXEC PGM=IEFBR14
//DD1       DD DSN=&UID..H?HOSTID..HSMPDOX,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//             VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
//DD2       DD DSN=&UID..H?HOSTID..HSMPDOY,DISP=(,CATLG),UNIT=?TRACEUNIT.,
//             VOL=SER=?TRACEVOL.,SPACE=(CYL,(20))
```

   If you have allocated these data sets as SMS-managed, they must be allocated on a specific volume and they must be associated with a storage class having the GUARANTEED SPACE attribute.

3. Allocate a generation data group (GDG) in which you can archive your site's trace history data.

   The following example defines the generation data group (GDG) name for the archived problem determination output data set. Substitute the applicable values you provided in step 1 of this work sheet, and run the following JCL job to create a generation data group.

```
//DEFGDG    JOB MSGLEVEL=1, TYPRUN=HOLD
//STEP1     EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=A
//SYSIN     DD *
  DEFINE    GDG(NAME('&UID..H?HOSTID..HSMTRACE') LIMIT(30) SCRATCH)
/*
```

4. Develop a procedure to automatically copy your PDA log data sets to tape.

   The following example shows you how to copy the inactive trace data set to tape as a generation data set (GDS). Substitute the applicable values you have provided in step 1 of this work sheet, and run the following JCL job to automatically copy your PDA log data sets to tape.

```
//PDOCOPY      JOB MSGLEVEL=1, TYPRUN=HOLD
//STEP1        EXEC PGM=IEBGENER
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  DUMMY
//SYSUT1   DD  DSN=&UID..H?HOSTID..HSMPDOY,DISP=SHR
//SYSUT2   DD  DSN=&UID..H?HOSTID..HSMTRACE(+1)
//             UNIT=TAPE,
//             DISP=(NEW,CATLG,CATLG),VOL=(,,1),
//             DCB=(&UID..H?HOSTID..HSMPDOY)
```

ARC1C159

Figure 103. Problem Determination Aid Log Data Set Size Work Sheet—Long-Term Trace History

Collection data set size work sheet

When using the DCOLLECT command or program ARCUTIL directly, if all (and only) DFSMShsm-specific records are requested, use the following work sheet (Figure 104) to calculate the size (in tracks) for the collection data set:

Collection Data Set Size Work Sheet

1. Fill in the blanks with values for your installation.

   ________ = MCDStracks - Number of tracks used by the MCDS.

   ________ = DCDStracks - Number of tracks used by the BCDS.

2. Substitute the values for *MCDStracks* and *BCDStracks* in the following calculation:

   (MCDStracks = .25 = ________) = ________________

   \+ (BCDStracks = .35 = ________) = ________________

Total = **total number of tracks for the collection data set**

ARC1C160

Figure 104. Collection Data Set Size Work Sheet

Appendix B. Accessibility

Accessible publications for this product are offered through IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

If you experience difficulty with the accessibility of any z/OS information, send a detailed message to the "Contact us" web page for z/OS (http://www.ibm.com/systems/z/os/zos/webqs.html) or use the following mailing address.

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
United States

Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features in z/OS can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

Consult assistive technologies

Assistive technology products such as screen readers function with the user interfaces found in z/OS. Consult the product information for the specific assistive technology product that is used to access z/OS interfaces.

Keyboard navigation of the user interface

You can access z/OS user interfaces with TSO/E or ISPF. The following information describes how to use TSO/E and ISPF, including the use of keyboard shortcuts and function keys (PF keys). Each guide includes the default settings for the PF keys.

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*
- *z/OS V2R2 ISPF User's Guide Vol I*

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users who access IBM Knowledge Center with a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line because they are considered a single compound syntax element.

Each line starts with a dotted decimal number; for example, `3` or `3.1` or `3.1.1`. To hear these numbers correctly, make sure that the screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol is placed next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol to provide information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, it indicates a reference that is defined elsewhere. The string that follows the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you must refer to separate syntax fragment OP1.

The following symbols are used next to the dotted decimal numbers.

? indicates an optional syntax element
The question mark (?) symbol indicates an optional syntax element. A dotted decimal number followed by the question mark symbol (?) indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that the syntax elements NOTIFY and UPDATE are optional. That is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

! indicates a default syntax element
The exclamation mark (!) symbol indicates a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the dotted decimal number can specify the ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the

default option for the FILE keyword. In the example, if you include the FILE keyword, but do not specify an option, the default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, the default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP applies only to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

*** indicates an optional syntax element that is repeatable**
The asterisk or glyph (*) symbol indicates a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3* , 3 HOST, 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you can write HOST STATE, but you cannot write HOST HOST.
3. The * symbol is equivalent to a loopback line in a railroad syntax diagram.

+ indicates a syntax element that must be included
The plus (+) symbol indicates a syntax element that must be included at least once. A dotted decimal number followed by the + symbol indicates that the syntax element must be included one or more times. That is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loopback line in a railroad syntax diagram.

Notices

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

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

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

COPYRIGHT LICENSE:

This information might contain 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (http://www.ibm.com/software/support/systemsz/lifecycle/)
- For information about currently-supported IBM hardware, contact your IBM representative.

Programming interface information

This document primarily documents information that is **not** intended to be used as a programming interface of DFSMShsm.

This document also documents intended programming interfaces that allow the customer to write programs to obtain the services of DFSMShsm. This information is identified where it occurs, either by an introductory statement or by the following marking:

Programming Interface Information

Programming interface information...

End Programming Interface Information

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at Copyright and Trademark information (http://www.ibm.com/legal/copytrade.shtml).

Index

Numerics

A

D

E

F

L

M

N

O

P

Q

R

S

T

U

V

W

Product Number: 5650-ZOS

Printed in USA

SC23-6869-02