

z/OS
3.2

MVS Setting Up a Sysplex



Note

Before using this information and the product it supports, read the information in [“Notices” on page 471.](#)

This edition applies to IBM® z/OS® 3.2 (5655-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2025-09-30

© **Copyright International Business Machines Corporation 1988, 2025.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures.....	xi
Tables.....	xv
About this information.....	xvii
Who should use this information.....	xvii
How to use this information.....	xvii
Where to find more information.....	xviii
How to provide feedback to IBM.....	xix
Summary of changes.....	xxi
Summary of changes for z/OS 3.2.....	xxi
Summary of changes for z/OS 3.1.....	xxi
Chapter 1. Introduction.....	1
Characteristics of a sysplex.....	1
Recommendations for redundancy.....	3
Coupling facility and the sysplex.....	3
Defining sysplex parameters and policies.....	4
Parmlib parameters and members.....	4
Policies.....	5
Using signaling for sysplex communications.....	6
Tuning the signaling services and coupling facility.....	6
Installing systems in a sysplex.....	7
Sharing a time source among LPARs.....	7
Chapter 2. Planning parmlib members for a sysplex.....	9
Planning the IEASYSxx member in parmlib.....	9
Considerations for the SYSNAME system parameter.....	9
Planning the PLEXCFG system parameter.....	9
Planning the GRS system parameter.....	11
Planning the GRSCNF system parameter.....	13
Planning the GRSRNL system parameter.....	13
Planning the CLOCK system parameter.....	14
Planning the COUPLE system parameter.....	17
Planning the IXGCNF system parameter.....	18
Planning the DRMODE System Parameter.....	18
Planning the COUPLExx member in parmlib.....	18
The COUPLE statement.....	19
The FUNCTIONS statement.....	19
CLASSDEF, PATHIN, PATHOUT, and LOCALMSG statements.....	21
The DATA statement.....	22
Modifying COUPLExx values.....	22
Planning the CONSOLxx member in parmlib.....	22
Naming your consoles.....	22
Understanding message routing.....	22
Related parmlib members to consider.....	22
Planning the CLOCKxx member in parmlib.....	23

Planning the XCFPOLxx member in parmlib.....	23
Planning the IEASYMxx member in parmlib.....	23
Specifying the SPINRCVY statement in the EXSPAT parmlib member.....	23
Preparing to use HyperPAV storage subsystems using the HYPERPAV statement in the IECIOSxx parmlib member.....	24
HyperPAV operations.....	24
Chapter 3. Planning the couple data sets.....	27
Considerations for all couple data sets.....	27
Considerations for a sysplex couple data set.....	32
Formatting a sysplex couple data set.....	32
Defining a sysplex couple data set to MVS.....	35
Connectivity requirements for a sysplex couple data set.....	35
Implications of the MAXSYSTEM value for the sysplex couple data set.....	36
Considerations for function couple data sets.....	37
Formatting couple data sets.....	37
Defining function couple data sets to MVS.....	37
Defining and activating policies.....	37
Updating policies in a sysplex.....	38
Connectivity requirements for function couple data sets.....	38
Providing coupling facility information to a recovery manager.....	39
Chapter 4. Managing coupling facility resources.....	41
Planning for a coupling facility.....	41
Defining a coupling facility.....	42
Managing a coupling facility.....	47
Planning a coupling facility policy.....	49
Identifying the coupling facility.....	49
Identifying the coupling facility structures.....	49
Comparing message-based processing and policy-based processing.....	68
An installation guide to duplexing rebuild.....	69
Understanding system-managed duplexing rebuild requirements.....	69
Deciding whether to duplex a structure.....	70
Migration steps.....	71
Migration steps for system-managed asynchronous duplexing	72
Controlling the use of the duplexing rebuild process.....	73
Policy changes affecting duplexing rebuild processes.....	73
Overview of the system-managed duplexing rebuild process.....	74
Common problems and solutions.....	78
Encrypting coupling facility structure data.....	80
The role of CFRM in a sysplex.....	84
Cleaning up status data.....	84
Performing system level initialization.....	84
Reconciling CFRM policy data.....	85
Transitioning to a new CFRM policy.....	86
Resource allocation in the coupling facility.....	87
Coupling facility storage considerations.....	87
Coupling facility resource allocation “rules”.....	87
Understanding coupling facility use of processor resources.....	88
Operating in a coupling facility environment.....	89
Removing a coupling facility from a sysplex.....	89
Transferring a coupling facility to another sysplex.....	90
Using REALLOCATE or POPULATECF to relocate coupling facility structures.....	90
Sample procedure for coupling facility maintenance.....	92
Summary of MVS operator commands.....	94
Coupling facility structures for IBM products.....	96

Chapter 5. Planning signaling services in a sysplex.....	99
Recommended steps for setting up signaling.....	99
Overview of available signaling options.....	100
Advantages of implementing signaling through list structures.....	102
Setting up outbound and inbound signaling paths.....	102
Implementing Signaling through CTC Connections.....	102
Implementing signaling paths through coupling facility list structures.....	102
Handling signaling path failures.....	104
Planning for optimum signaling performance.....	105
XCF groups.....	106
Planning the transport classes.....	106
Planning the message buffer space.....	112
Adding and deleting signaling paths.....	115
Adding a signaling path.....	115
Deleting a signaling path.....	116
Sample signaling scenarios.....	117
Example 1 — Using COUPLExx defaults.....	117
Example 2 — Creating a transport class for large messages.....	119
Example 3 — Creating a transport class for a group.....	121
Example 4 — Defining signaling paths through the coupling facility.....	123
Example 5 — Signaling through a coupling facility and CTC connections.....	125
Handling signalling path problems	127
Problems with signalling path definitions.....	128
Loss of some signalling paths between systems.....	129
Loss of all signalling paths between some systems.....	130
Loss of all signalling paths between all systems.....	130
Signaling sympathy sickness.....	130
WSC flash — Parallel Sysplex performance: XCF performance considerations.....	131
Transport classes.....	131
Message Buffers.....	131
Signaling paths.....	134
Case study.....	138
 Chapter 6. Planning XCF Note Pad Services in a sysplex.....	 141
Steps for setting up XCF Note Pad Services.....	141
Defining the XCF note pad catalog structure in the CFRM policy.....	141
Name of the XCF note pad catalog structure.....	141
Determining the size of the XCF note pad catalog structure.....	141
Other considerations for the XCF note pad catalog structure.....	143
Deleting the XCF note pad catalog structure.....	143
Defining XCF note pad structures in the CFRM policy.....	144
Naming conventions for the XCF note pad structures.....	144
Determining the sizes of the XCF note pad structures.....	145
Other considerations for the XCF note pad structures.....	146
Deleting XCF note pad structures.....	146
Authorizing XCF note pad requests.....	147
Restricting IXLCONN access to XCF catalog and note pad structures.....	148
 Chapter 7. Tuning a sysplex.....	 149
RMF reports and the sysplex.....	149
XCF activity report.....	149
Tuning the signaling service.....	149
Importance of Adequate CPU Resources.....	150
Tuning the message buffer size (class length).....	151
Tuning the maximum message buffer space.....	154
Tuning the signaling paths.....	158

Tuning the transport classes.....	160
Capacity Planning.....	162
RMF XCF Activity Report — Sample.....	164
Tuning UNIX System Services performance in a sysplex.....	166
Tuning coupling facilities.....	166
Coupling facility activity report.....	167
SYNC and ASYNC service.....	171
CHNGD requests.....	172
Potential sources of change and delay in service times.....	172
Chapter 8. Planning sysplex availability and recovery.....	177
Controlling availability and recovery through COUPLExx.....	178
Planning the failure detection interval and operator notification interval.....	178
Planning the cleanup interval.....	179
Controlling system availability and recovery through the SFM Policy.....	180
Overview and requirements of SFM policy.....	180
Planning for a status update missing condition.....	181
Handling signaling connectivity failures.....	183
Handling coupling facility connectivity failures.....	185
Handling signaling sympathy sickness.....	185
Planning PR/SM reconfigurations.....	186
Setting Up an SFM policy.....	187
Controlling availability and recovery through XCFPOLxx (PR/SM only).....	187
Implementing XCFPOLxx.....	188
Examples.....	188
Example 1: Failure management in a PR/SM environment — one processor.....	188
Example 2: Failure management in a PR/SM environment — two processors and a coupling facility.....	189
Example 3: Failure management in a PR/SM environment — one processor.....	191
Example 4: Failure management with SFM policy and a coupling facility.....	191
Controlling job availability and recovery through automatic restart management.....	193
Requirements for participating in automatic restart management.....	193
Using automatic restart management.....	193
Using the system status detection partitioning protocol and BCPii for availability and recovery	196
Handling concurrent system and couple data set failures.....	197
Planning disaster recovery actions.....	197
Chapter 9. Adding MVS systems to a sysplex.....	201
XCF-local mode configuration.....	201
Single-system sysplex configuration.....	202
Multisystem sysplex configuration without an existing global resource serialization complex.....	203
Multisystem sysplex configuration with an global resource serialization complex.....	204
The first system IPL.....	205
The second system IPL.....	206
The third system IPL.....	206
The fourth system IPL.....	207
Multisystem sysplex configuration on one processor under z/VM.....	208
Multisystem sysplex configuration on PR/SM.....	209
Multisystem sysplex configuration with a coupling facility.....	209
Multisystem sysplex configuration of greater than eight systems.....	210
Removing a system from the sysplex.....	210
Lock structure considerations.....	211
Multisystem application considerations.....	211
Rejoining a sysplex.....	212
Chapter 10. Planning for system logger applications.....	213
What is system logger?.....	213

Coupling facility log stream.....	214
DASD-only log stream.....	215
The system logger configuration.....	215
The system logger component.....	217
The LOGR couple data set (LOGR policy).....	218
The system logger single-system scope couple data sets (LOGRY and LOGRZ) policies.....	219
Log data on the coupling facility.....	220
Log data on DASD log data sets.....	220
Duplexing log data.....	221
Offloading log data from interim storage by freeing and/or moving it to DASD.....	222
System clock considerations.....	230
Logger configuration resource value ranges/limits.....	231
Considerations for encrypting log stream data sets.....	236
Finding information for system logger applications.....	239
Finding information for CICS log manager.....	239
Finding information for OPERLOG log stream.....	240
Finding information for logrec log stream.....	241
Finding information for APPC/MVS.....	242
Finding information for IMS common queue server log manager.....	242
Finding information for resource recovery services.....	243
Finding information for system management facilities (SMF).....	243
Preparing to use system logger applications.....	243
Understand the requirements for system logger.....	244
Plan the system logger configuration.....	247
Determine the size of each coupling facility structure.....	255
Develop a naming convention for system logger resources.....	258
Plan DASD space for system logger.....	260
Managing log data: How much? For how long?.....	272
Define authorization to system logger resources.....	273
Format the sysplex scope LOGR couple data set and make it available to the sysplex.....	276
Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex.....	279
Add information about log streams and coupling facility structures to the LOGR policy.....	280
Deleting log streams from the LOGR, LOGRY or LOGRZ policy.....	289
Define the coupling facility structures attributes in the CFRM function couple data set.....	289
Activate the LOGR subsystem.....	291
Deleting log data and log data sets.....	291
Deleting log data.....	292
Deleting log data sets.....	292
When is my log data or log data set physically deleted?.....	292
Finding and deleting orphaned log data sets.....	293
Upgrading an existing log stream configuration.....	293
Changing the GROUP value.....	293
Upgrading a log stream from DASD-only to coupling facility.....	295
Updating a log stream to use a different coupling facility structure.....	296
Updating a log stream's attributes.....	296
Renaming a log stream dynamically	299
System logger recovery.....	300
Operations log and logrec application recovery.....	300
Recovery performed for DASD-only log streams.....	300
When an MVS system fails.....	301
When a sysplex fails.....	301
When the system logger address space fails or hangs.....	302
When the coupling facility structure fails.....	305
When the coupling facility space for a log stream becomes full.....	308
When a staging data set becomes full.....	308
When a log stream is damaged.....	309
When DASD log data set space fills.....	309
When unrecoverable DASD I/O errors occur.....	310

Preparing for z/OS IBM zAware log stream client usage.....	311
Chapter 11. Format utility for couple data sets.....	313
Using the couple data set format utility.....	313
Understanding couple data set coexistence.....	313
Coding the couple data set format utility.....	314
Sysplex parameters for format utility.....	317
Automatic restart management parameters for format utility.....	318
CFRM parameters for format utility.....	319
LOGR parameters for format utility (sysplex scope).....	320
LOGRY or LOGRZ couple data set versioning - new format levels.....	327
Formatting a sysplex scope LOGR couple data set.....	327
Formatting a single-system scope LOGRY or LOGRZ couple data set.....	328
SFM parameters for format utility.....	328
z/OS UNIX parameters for format utility.....	328
BPXMCDS couple data set versioning — new format level.....	329
WLM parameters for format utility.....	329
Return codes.....	330
Examples of using the IXCL1DSU format utility.....	331
Formatting a sysplex couple data set.....	331
Formatting an automatic restart management couple data set.....	331
Formatting a CFRM couple data set.....	331
Formatting a sysplex scope LOGR couple data set.....	331
Formatting an SFM couple data set.....	332
Formatting a UNIX System Services couple data set.....	332
Formatting a WLM couple data set.....	332
Sample output.....	332
Chapter 12. Administrative data utility.....	335
Using the administrative data utility.....	335
Authorizing use of the utility.....	335
Coding the administrative data utility.....	336
Automatic restart management parameters for administrative data utility.....	338
CFRM parameters for administrative data utility.....	346
LOGR keywords and parameters for the administrative data utility.....	358
LOGRY and LOGRZ keywords and parameters for the administrative data utility.....	386
SFM parameters for the administrative data utility.....	392
Return codes.....	397
Examples of using the IXCMIAPU utility.....	397
Administrative data utility for automatic restart manager Policy Data.....	397
Administrative data utility for CFRM policy data.....	398
Administrative data utility for the LOGR policy.....	398
Administrative data utility for the LOGRY and LOGRZ policy data.....	399
Administrative data utility for SFM policy data.....	399
Chapter 13. Deletion utility for XCF group members.....	401
Output messages from the IXCDELET utility.....	401
Chapter 14. Deletion utility for XCF note pads.....	403
Output messages from the IXCDELNP utility.....	403
Chapter 15. Calculating sizes for signaling services.....	407
Calculating the size of a list structure for signaling.....	407
Additional considerations.....	407
Message information for some MVS groups.....	409
Calculating message buffer space — An example.....	410
Calculating the size of list structures for system logger applications.....	412

Chapter 16. Sysplex configurations.....	419
Under VM - Physical configuration of a sysplex with MVS Guests.....	419
Components of the z/VM Guest Coupling Simulation Support.....	419
Setting up a guest coupling environment under z/VM.....	420
On PR/SM - Physical configuration of a sysplex on LPARs.....	423
Configuration of a sysplex on LPAR and Non-LPAR systems.....	424
Chapter 17. Coupling Facility Guidelines.....	425
A Review of Some Basic Concepts.....	425
A Review of the Use of Structure Data.....	425
A Review of the Concept of Persistence.....	425
Guidelines for coupling facility reconfiguration.....	427
Adding a Coupling Facility.....	429
Adding a structure.....	430
Planning for a Coupling Facility Shutdown.....	430
Shutting down a coupling facility for maintenance.....	431
Removing a coupling facility from the configuration.....	432
Upgrading or replacing a coupling facility.....	433
Upgrading or replacing all coupling facilities	434
Transitioning from a Parallel Sysplex to a base sysplex.....	436
Removing structures from a coupling facility for shutdown.....	437
Removing a structure from a coupling facility.....	438
Deleting a structure from the CFRM active policy.....	440
Structure rebuild recovery.....	448
Guidelines for unplanned outages.....	449
Guidelines for recovering from coupling facility failures.....	449
Handling failing or disconnecting connection states.....	452
Best practices for updating a coupling facility.....	456
"Push/pull" of a coupling facility or POR of a processor with a coupling facility.....	457
POR of a CPC with a Coupling Facility with no Physical or Logical Changes to the CF.....	461
Disruptive CF Upgrade.....	464
Chapter 18. Using automatic tape switching (ATS STAR).....	467
Setting up automatic tape switching.....	467
Recommendations for automatic tape switching.....	468
Appendix A. Accessibility.....	469
Notices.....	471
Terms and conditions for product documentation.....	472
IBM Online Privacy Statement.....	473
Policy for unsupported hardware.....	473
Minimum supported hardware.....	473
Trademarks.....	474
Glossary.....	475
Index.....	481

Figures

1. Sysplex with Coupling Facility and Couple Data Sets.....	3
2. Example: Results from D M=DEV command.....	25
3. Example: Results from D IOS,HYPERPAV command.....	25
4. Signaling on a Four-System Sysplex, Implemented through CTC Connections.....	100
5. Example of SYSA's PATHIN and PATHOUT Specifications in COUPLExx.....	101
6. Signaling on a Four-System Sysplex, Implemented through a Coupling Facility List Structure.....	101
7. Example of SYSW's PATHIN and PATHOUT Specifications in COUPLExx.....	101
8. Syntax for CLASSDEF Statement of COUPLExx parmlib Member.....	108
9. Syntax for PATHIN, PATHOUT, LOCALMSG on COUPLExx parmlib Member.....	112
10. SYSA with Inbound, Local, and Outbound message Buffers.....	114
11. Example 1 — Diagram of PLEXOF2.....	118
12. Example 1 — SYSA's COUPLExx Parmlib Member.....	118
13. Example of SYSB's COUPLExx Parmlib Member.....	119
14. Example 2 — Diagram of PLEXOF2B.....	120
15. Example 2 — SYSA's COUPLExx Parmlib Member.....	120
16. Example 3 — Diagram of PLEXOF2A.....	122
17. Example of SYSA's COUPLExx Parmlib Member.....	122
18. Example 4 — Diagram of PLEXOF2C.....	124
19. Example 4 — SYSA's COUPLExx Parmlib Member.....	124
20. Example 5 — Diagram of PLEXOF2D.....	126
21. Example of SYSA's COUPLExx Parmlib Member.....	126
22. Example: RMF XCF Report.....	132
23. Example: XCF Path Statistics.....	133

24. Example: XCF Usage By System - Outbound Messages.....	134
25. Example: XCF Usage By System - Inbound Messages.....	134
26. Example: XCF Usage by System - ALL PATH UNAVAIL.....	135
27. Example: XCF Path Statistics - Requests Routed through DEFAULT Class.....	135
28. Example: Display XCF Command Path Response Time.....	136
29. Example: XCF Path Statistics.....	137
30. Example: RMF Report - XCF Delays.....	138
31. Example: RMF Report - Defining New Transport Class.....	138
32. Example: RMF Report - Increasing CLASSLEN.....	139
33. Example: Insufficient Message Buffer Space for Outbound Path.....	139
34. Example: Increasing the MAXMSG on the PATHOUT.....	139
35. Example: BUSY Conditions Reduced.....	140
36. XCF Activity Report - Usage by System.....	164
37. XCF Activity Report - Usage by Member.....	165
38. XCF Activity Report - Path Statistics.....	166
39. CF synchronous service time.....	170
40. Example: RMF CF Structure Report.....	171
41. Example: RMF CF Usage Report.....	173
42. Example: RMF CF Subchannel Activity Report.....	175
43. Three-System Sysplex with Active SFM Policy.....	183
44. Signaling Connectivity Failure Between SYSA and SYSB.....	184
45. Three-System Sysplex on Two Processors with Active SFM Policy.....	187
46. Failure Management in a PR/SM Environment — One Processor.....	188
47. Failure Management in a PR/SM Environment — Two Processors and a Coupling Facility.....	190
48. Failure Management in a PR/SM Environment — One Processor.....	191

49. Failure Management with SFM Policy and a Coupling Facility.....	192
50. Example: specifying the CFRMPOL keyword in COUPLExx parmlib member.....	199
51. XCF-Local Mode Configuration.....	202
52. Configuration with a Single-System Sysplex and Three Independent Systems.....	203
53. Multisystem Sysplex Configuration.....	204
54. One System in the Multisystem Sysplex.....	205
55. Two Systems in the Multisystem Sysplex.....	206
56. Three Systems in the Multisystem Sysplex.....	207
57. Four Systems in the Multisystem Sysplex.....	207
58. Multisystem Sysplex Configuration with a Coupling Facility.....	210
59. Log Stream Data on the Coupling Facility and DASD.....	214
60. Log Stream Data in Local Storage Buffers and DASD Log Data Sets.....	215
61. A Complete Coupling Facility Log Stream Configuration.....	216
62. A DASD-Only Configuration.....	216
63. Example: Merging Log Data from Systems Across a Sysplex.....	250
64. Example: Maintaining Multiple Log Streams in the Same Coupling Facility Structure.....	251
65. Example: Recovery for a Failing System - All Data Recovered.....	252
66. Example: Recovery for a Failing System - Recovery Delayed for Some Data.....	253
67. Example: Using Staging Data Sets When the Coupling Facility Shares a CPC with a System.....	285
68. Example: Using Staging Data Sets When the Coupling Facility is Volatile.....	286
69. Example of formatting a primary and alternate couple data set for the LOGR policy.....	328
70. Example of formatting a primary and alternate couple data set for the LOGR policy.....	331
71. Sample Output of Couple Data Set Format Utility.....	333
72. FACILITY Class Profiles.....	335
73. Syntax of CFRM parameters for the Administrative Data Utility.....	346

74. Example: Using * to Assign Installation Defaults in SFM Policy.....	396
75. IXCCFRMP - Administrative Policy Data in CFRM Couple Data Set.....	398
76. Example of running the utility to create or update the LOGR administrative policy.....	399
77. Example of running the utility to create or update the LOGR administrative policy.....	399
78. IXCSFMP - Administrative Policy Data in SFM Couple Data Set.....	400
79. Example: JCL to define the CF Service Machines to MVS.....	421
80. Sample: definition of CLOCKxx member for z/VM.....	422
81. On PR/SM - Physical Configuration of a Sysplex on LPARs.....	423
82. Configuration of a Sysplex on LPAR and Non-LPAR Systems.....	424
83. Message Output from a Rebuild Hang.....	449
84. Example: Displaying information about an IRLM lock structure.....	453
85. Example: Output from DISPLAY XCF, STRUCTURE, STRNAME command.....	455

Tables

1. XCF/XES Optional Functions	19
2. CFLEVEL summary table.....	44
3. Initial Structure Size Allocation.....	51
4. Common System-Managed Duplexing Rebuild Problems.....	78
5. Comparison of POPULATECF and REALLOCATE.....	91
6. Summary of MVS commands for managing a coupling facility.....	94
7. Coupling facility structures in z/OS for IBM products.....	96
8. MAXMSG Values on COUPLE Statements.....	113
9. Sample catalog structure sizes for different note pad counts.....	141
10. Sample note pad structure sizes for different note counts.....	145
11. Summary of Policies in Effect.....	181
12. Statements and Keywords for Initiating PR/SM Reconfigurations.....	188
13. Automatic Restart Management Parameters for IXCMIAPU and IXCARM Parameter Overrides.....	194
14. Summary of logger resources.....	231
15. Finding CICS log manager information.....	240
16. Finding OPERLOG information.....	240
17. Finding Logrec Log Stream information.....	241
18. Finding APPC/MVS information.....	242
19. Finding IMS (CQS) log manager information.....	243
20. Reviewing System Logger Setup Activities.....	244
21. Settings to consider for data sets approaching 4GB - z/OS Version 1 Release 11.....	271
22. Settings to consider for data sets approaching 4GB - z/OS Version 1 Release 13.....	271
23. Logger and System-Managed Duplexing Rebuild Combinations.....	287

24. System Logger logstream attribute dynamic update commit outline.....	297
25. Support for CFRM functions.....	319
26. LOGR CDS format levels.....	322
27. Considerations for supporting key system logger functions.....	322
28. LOGRY and LOGRZ CDS format levels.....	327
29. Considerations for supporting key system logger functions.....	327
30. SCMALGORITHM value descriptions.....	351
31. DUPLEX keywords.....	355
32. Valid special (graphical) characters.....	370
33. Valid special (graphical) characters.....	383
34. DEFINE LOGSTREAM keywords and parameters for single-system scope policy.....	388
35. UPDATE LOGSTREAM keywords and parameters for single-system scope policy.....	390
36. DELETE LOGSTREAM keywords and parameters for single-system scope policy.....	391
37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides.....	457
38. Steps: POR of a CPC with a coupling facility with no physical or logical changes to the CF.....	461
39. Steps: move a disrupted coupling facility.....	464

About this information

This information describes how to set up a z/OS system to run in a Parallel Sysplex®.

The information assumes that you have a good knowledge of MVS and that you are familiar with the enhanced capabilities of data sharing and parallel processing presented in the z/OS Parallel Sysplex Library.

Who should use this information

System programmers should use this information for information that they need to plan, install, operate, modify, and tune a z/OS system in a sysplex.

How to use this information

This information is organized as follows:

Chapter 1, “Introduction,” on page 1 — describes the characteristics of a sysplex, introduces basic sysplex concepts and provides an overview of the topics covered in this information.

Chapter 2, “Planning parmlib members for a sysplex,” on page 9 — provides an overview of the SYS1.PARMLIB members that you need to set up a sysplex.

Chapter 3, “Planning the couple data sets,” on page 27 — describes how to plan couple data sets for the sysplex.

Chapter 4, “Managing coupling facility resources,” on page 41 — provides an overview of the coupling facility and contains information on planning and setting up a coupling facility resource management policy.

Chapter 5, “Planning signaling services in a sysplex,” on page 99 — describes MVS signaling to communicate in the sysplex.

Chapter 7, “Tuning a sysplex,” on page 149 — contains tuning information for a sysplex.

Chapter 8, “Planning sysplex availability and recovery,” on page 177 — describes availability and recovery in a sysplex.

Chapter 9, “Adding MVS systems to a sysplex,” on page 201 — illustrates various sysplex configurations.

Chapter 10, “Planning for system logger applications,” on page 213 — contains information for planning a system logger application.

Chapter 11, “Format utility for couple data sets,” on page 313 — contains the information needed to format couple data sets.

Chapter 12, “Administrative data utility,” on page 335 — contains the information needed to define policies.

Chapter 13, “Deletion utility for XCF group members,” on page 401 — contains the information needed to delete XCF group members.

Chapter 15, “Calculating sizes for signaling services,” on page 407 — contains formulas for calculating the size of coupling facility list structures and message buffers for your sysplex.

Chapter 16, “Sysplex configurations,” on page 419 — contains diagrams of some typical sysplex configurations.

Chapter 17, “Coupling Facility Guidelines,” on page 425 — contains guidance and procedures for changing the configuration of a coupling facility in a sysplex.

Chapter 18, “Using automatic tape switching (ATS STAR),” on page 467 — contains information about automatically switchable tape devices.

“Notices” on page 471 — contains the Notices and Trademarks list.

This information also contains a glossary.

Where to find more information

This information references information in other publications, using shortened versions of the information titles. For complete titles and order numbers of the books for z/OS products, see [*z/OS Information Roadmap*](#).

See IBM z/OS Parallel Sysplex (www.ibm.com/products/zos/parallel-sysplex) for links to additional information related to sysplex processing.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. For more information, see [How to send feedback to IBM](#).

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.

Note: IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under [IBM z/OS Product Documentation Update Policy](http://www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy) (www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy).

Summary of changes for z/OS 3.2

The following content is new, changed, or no longer included in z/OS 3.2.

New

The following content is new.

September 2025 release

- None.

Changed

The following content is changed.

September 2025 release

- [“CFRM parameters for administrative data utility” on page 346](#) is updated.
- [“Considerations for supporting CFRM functions” on page 319](#) is updated.
- [“CFRM parameters for format utility” on page 319](#) is updated.
- [“What is system logger?” on page 213](#) is updated.

Deleted

The following content is deleted.

September 2025 release

- None.

Summary of changes for z/OS 3.1

The following content is new, changed, or no longer included in z/OS 3.1.

Changed

The following content is changed.

September 2024 refresh

- [“Comparing message-based processing and policy-based processing” on page 68](#) is updated with information for changing the manager system for MSGBASED processing.

June 2024 refresh

- [“CFRM parameters for format utility” on page 319](#) and [“Considerations for supporting CFRM functions” on page 319](#) are updated to correct the maximum number of CFRM structures that a couple data set can be formatted for as described by APAR [OA62794: REVERT TO MAX 2048](#)

STRUCTURES IN CFRM POLICY (www.ibm.com/support/pages/apar/OA62794) for z/OS V2R3 and newer releases.

November 2023 refresh

- “Other events that trigger offloading” on page 225 is updated with information for the LOCALBUFFERLIMIT option for parmlib member IXGCNFxx. (APAR [OA64676](http://www.ibm.com/support/pages/apar/OA64676): NEW FUNCTION - LOCALBUFFERLIMIT SUPPORT (www.ibm.com/support/pages/apar/OA64676), applies to z/OS V2R5 and newer)

Chapter 1. Introduction

A **sysplex** is a collection of z/OS systems that cooperate, using certain hardware and software products, to process workloads. The products that make up a sysplex cooperate to provide higher availability, easier systems management, and improved growth potential over a conventional computer system of comparable processing power.

The cross system coupling facility (XCF) component of z/OS provides simplified multisystem management. XCF services allow authorized programs on one system to communicate with programs on the same system or on other systems. If a system fails, XCF services also provide the capability for batch jobs and started tasks to be restarted on another eligible system in the sysplex.

Server Time Protocol (STP) or a Sysplex Timer is required to synchronize the time-of-day (TOD) clocks for systems in a sysplex that run on different processors. Note, however, that Sysplex Timer connections are no longer supported starting with the z196/z114 servers.

In a Parallel Sysplex, central processing complexes (CPCs) are connected through channel-to-channel (CTC) communications or through a *coupling facility*.

A coupling facility enables parallel processing and improved data sharing for authorized programs running in the sysplex. The cross-system extended services (XES) component of MVS enables applications and subsystems to take advantage of the coupling facility.

A shared sysplex couple data set records status information for the sysplex.

Characteristics of a sysplex

A sysplex can include the following software and hardware:

- z/OS

These products include the cross-system coupling facility (XCF) component, which enables authorized programs in a sysplex to communicate with programs on the same MVS system or other MVS systems; and the global resource serialization component, which serializes sysplex resources.

z/OS provides support for the coupling facility, which provides high-speed access to shared data across applications and subsystems running on different MVS systems.

- Signaling paths between MVS systems

There must be at least two operational signaling paths (one inbound and one outbound path) between each of the MVS systems in the sysplex. The signaling paths can be defined through:

- Coupling facility list structures
- ESCON or FICON® channels operating in CTC mode
- 3088 Multisystem Channel Communication Unit

- Sysplex couple data set

MVS requires a DASD data set (and an alternate data set is recommended for availability) to be shared by all systems in the sysplex. On the sysplex couple data set, MVS stores information related to the sysplex, systems, XCF groups, and their members. An XCF group is the set of related members that a multisystem application defines to XCF. A multisystem application can be an installation-defined program, an MVS component or subsystem, or a program product.

A sysplex couple data set is always required for a multisystem sysplex and for most single-system sysplexes. However, you can define a single system sysplex that does not require a sysplex couple data set.

- Common time reference

When the sysplex consists of multiple MVS systems running on two or more processors, MVS requires that the processors be connected to the same time source to synchronize TOD clocks across CPCs. This can be done by using a common 9037 Sysplex Timer or through STP. Starting with the z114/z196 servers, connections to the 9037 Sysplex Timer are no longer supported. MVS uses the Sysplex Timer to synchronize TOD clocks across systems.

For a multisystem sysplex defined on a single processor (under PR/SM or VM) the SIMETRID parameter in the CLOCKxx parmlib member must specify the simulated Sysplex Timer identifier to synchronize timings for the MVS systems. When a sysplex with LPARs resides on two or more physical CECs, it requires ETRMODE YES or STPMODE YES to be defined in the CLOCKxx parmlib member.

To manage certain aspects of your sysplex, you can install one or more additional couple data sets, for:

- Coupling facility resource management (CFRM), to define how the system is to manage coupling facility resources.
- Sysplex failure management (SFM), to define how the system is to manage system and signaling connectivity failures and PR/SM reconfiguration actions.
- Workload management (WLM), to define service goals for workloads.
- Automatic restart management (ARM), to define how to process restarts for started tasks and batch jobs that have registered with automatic restart management.
- System logger (LOGR, LOGRY and LOGRZ), to define, update, or delete structure or log stream definitions.
- z/OS UNIX System Services (z/OS UNIX), to contain file information when using a shared file system in a sysplex.

Figure 1 on [page 3](#) shows a sysplex of two z/OS systems, SYSA and SYSB, where both systems are connected to a Sysplex Timer and a coupling facility and can access the sysplex couple data set and the CFRM couple data set. The sysplex might also be configured so the systems can access one or more couple data sets, such as ARM (automatic restart management), LOGR, SFM, WLM, or z/OS UNIX.

(For diagrams of additional configurations, see Chapter 9, “Adding MVS systems to a sysplex,” on [page 201](#) and Chapter 16, “Sysplex configurations,” on [page 419](#).)

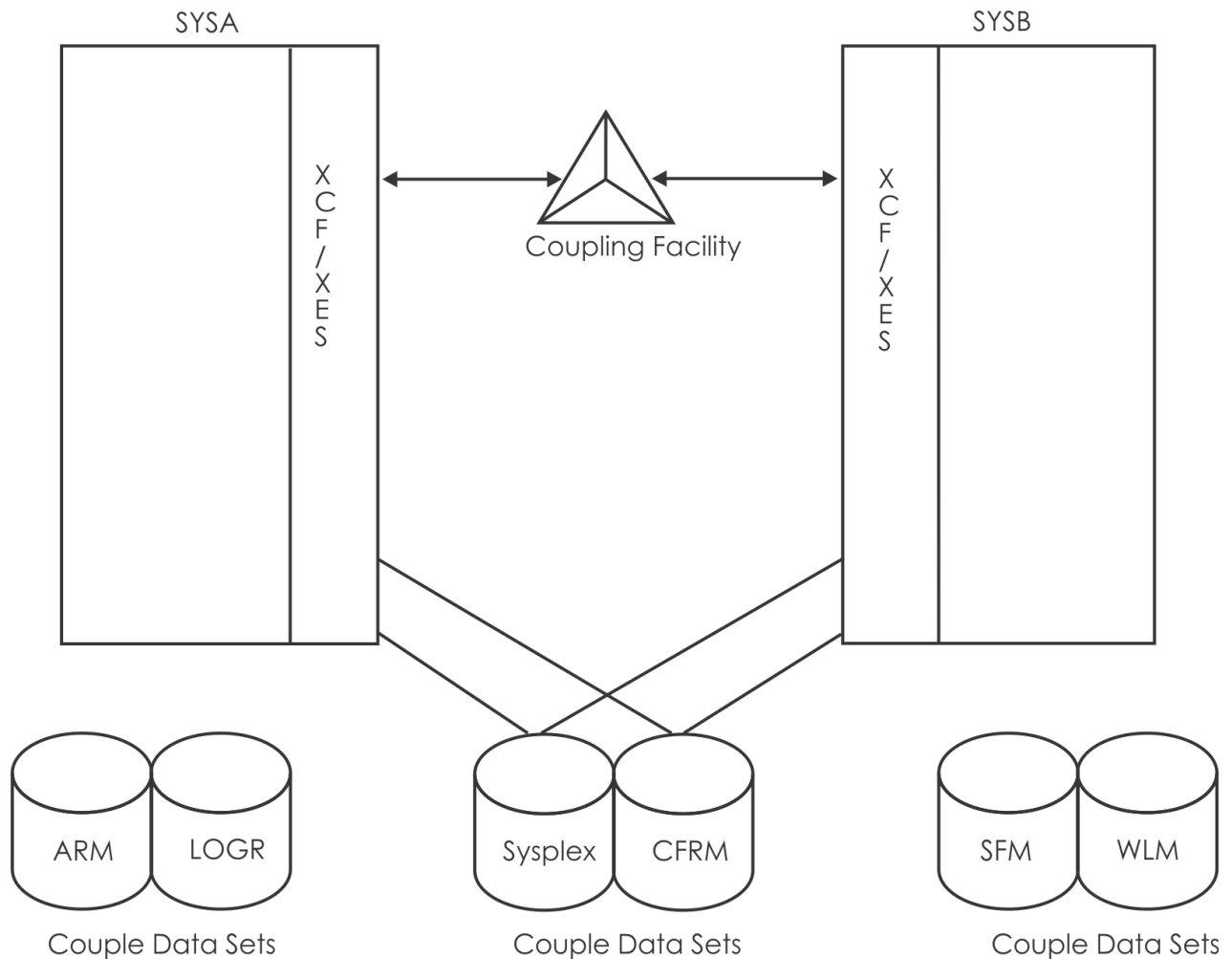


Figure 1. Sysplex with Coupling Facility and Couple Data Sets

Recommendations for redundancy

IBM strongly recommends that you install redundant hardware in the sysplex, including the hardware for signaling paths, the Coupling links, and the coupling facility. In addition, IBM recommends redundancy for the sysplex couple data set and for any other couple data sets that you use.

Coupling facility and the sysplex

Cross-system extended services (XES) uses one or more coupling facilities to satisfy customer requirements for:

- Data sharing across the systems in a sysplex
- Maintaining the integrity and consistency of shared data
- Maintaining the availability of a sysplex.

A coupling facility is a special logical partition on IBM Z® and zEnterprise® servers. The coupling facility allows software on different systems in the sysplex to share data. To share data, systems must have connectivity to the coupling facility through coupling facility links.

Systems in the sysplex that are using a coupling facility must also be able to access the coupling facility resource management (CFRM) couple data set. [Chapter 4, “Managing coupling facility resources,” on page 41](#) describes the coupling facility in detail.

Defining sysplex parameters and policies

The values you specify in parmlib largely control the characteristics of systems in a sysplex.

Policies allow MVS to manage specific resources in conformance with your system and resource requirements and with little operator intervention.

Parmlib parameters and members

Many of the values that represent the fundamental decisions you make about the systems in your sysplex are in parmlib. The following list describes the key parmlib members you need to consider when setting up an MVS system to run in a sysplex:

- IEASYSxx is the system parameter list, which holds parameter values that control the initialization of MVS. Some parameters identify other members of parmlib that are to be used. (For example, the GRSCNF system parameter identifies the GRSCNFxx member.)
- COUPLExx contains parameter values that reflect sysplex-related information.
- GRSCNFxx describes the global resource serialization complex for the system.
- GRSRNLxx specifies the resource name lists to be used to control the serialization of resources in the complex.
- CLOCKxx indicates how the time of day is to be set on the system.
- IEASYMxx provides a single place to define system parameters and system symbols for all systems in a sysplex.
- LOADxx optionally specifies the following:
 - The name of a sysplex in which a system participates, which is also the substitution text for the &SYSPLEX system symbol
 - One or more IEASYMxx parmlib members that specify the system symbols and system parameters to be used.
- XCFPOLxx specifications are used only if a sysplex failure management policy is not active in the sysplex. Functions provided through XCFPOLxx can provide high availability for multisystem applications on an MVS system on PR/SM. If SFM is active in the sysplex, then the SFM policy specifications can define the same PR/SM reconfiguration actions that can be defined through XCFPOLxx.
- CONSOLxx contains values that define your console configuration.
- IXGCNFxx contains values that define your system logger configuration.

Sharing parmlib members

To prevent the number of parmlib definitions from increasing with each system that is added to a sysplex environment, you can share parmlib members across systems. If systems require unique values in those parmlib members, you can code them so the different systems can supply unique values in shared parmlib definitions.

System symbols are the elements that allow systems to specify unique values in shared parmlib members. When specified in a parmlib definition that two or more system images are to share, system symbols resolve to unique values on each system.

Additional information

Background information and considerations for selecting values for parmlib members are presented in Chapter 2, “Planning parmlib members for a sysplex,” on [page 9](#) in the following topics:

- [“Planning the IEASYSxx member in parmlib” on page 9](#)
- [“Planning the COUPLExx member in parmlib” on page 18](#)
- [“Planning the XCFPOLxx member in parmlib” on page 23](#)

- [“Planning the IEASYMxx member in parmlib” on page 23.](#)

Policies

A policy is a set of rules and actions that systems in a sysplex are to follow when using certain MVS services. A policy allows MVS to manage specific resources in compliance with your system and resource requirements, but with little operator intervention. A policy can be set up to govern all systems in the sysplex or only selected systems. You might need to define more than one policy to allow for varying workloads, configurations, or other installation requirements at different times. For example, you might need to define one policy for your prime shift operations and another policy for other times. Although you can define more than one policy of each type (except for system logger) only one policy of each type can be active at a time. For system logger, there is only one sysplex scope LOGR policy in the sysplex, and up to two (2) distinct single-system scope policies LOGRY and LOGRZ.

The following policies can be used to enhance systems management in a sysplex:

- The coupling facility resource management (CFRM) policy allows you to define how MVS is to manage coupling facility resources.
- The sysplex failure management (SFM) policy, allows you to define how MVS is to manage system failures, signaling connectivity failures, and PR/SM reconfiguration actions.
- The workload management (WLM) policy allows you to define service goals for workloads.
- The automatic restart management policy allows you to define how MVS is to manage automatic restarts of started tasks and batch jobs that are registered as elements of automatic restart management.
- The system logger (LOGR) policy allows you to define, update, or delete structure or log stream definitions. The system logger (LOGRY and LOGRZ) policies allow you to define, update or delete DASD-only type log stream definitions.

Setting up couple data sets for policy data

Each of these policies resides in a couple data set, which you need to format and define to MVS. The different types of policies can all reside in the same couple data set, or they can reside in separate couple data sets, either on the same volume or on different volumes. To make a policy active on an MVS system, you must define the policy and activate it.

IBM provides a couple data set format utility, IXCL1DSU in SYS1.MIGLIB, which you can use to format the couple data sets. Sample JCL to invoke this utility and an explanation of the information you need to specify when formatting a couple data set is included in [Chapter 11, “Format utility for couple data sets,” on page 313](#). [Chapter 3, “Planning the couple data sets,” on page 27](#) includes an explanation of the format utility and a discussion of what you need to consider when formatting a couple data set. (The couple data set for WLM policy data can also be formatted interactively using the administrative application. See [z/OS MVS Planning: Workload Management](#).)

Defining policies

Once the couple data sets for a service are formatted, you can define the policies that are to reside in the data sets. These policies are called administrative policies.

For the CFRM, SFM, LOGR, LOGRY, LOGRZ and automatic restart management administrative policies, IBM supplies an administrative data utility, IXCMIAPU, in SYS1.MIGLIB. You can use this utility to create new policies, delete policies that are no longer required, and provide reports describing the policies. [Chapter 12, “Administrative data utility,” on page 335](#) describes the utility and provides examples for its use.

To work with WLM administrative policies, you can use the administrative application, which is described in [z/OS MVS Planning: Workload Management](#).

You can find information on defining and activating policies as follows:

- For the CFRM policy, see [Chapter 4, “Managing coupling facility resources,” on page 41](#).

- For the SFM policy, see Chapter 8, “Planning sysplex availability and recovery,” on page 177.
- For the LOGR policy, see [“LOGR keywords and parameters for the administrative data utility”](#) on page 358.
- For the LOGRY and LOGRZ policies, see [“LOGRY and LOGRZ keywords and parameters for the administrative data utility”](#) on page 386.
- For the WLM policy, see *z/OS MVS Planning: Workload Management*.
- For the automatic restart management policy, see [“Controlling job availability and recovery through automatic restart management”](#) on page 193.

Setting up couple data sets for z/OS UNIX

The z/OS UNIX System Services element uses a couple data set to maintain information to support a shared file system in a sysplex. There is no policy associated with the OMVS couple data set, although it must be formatted with the IXCL1DSU format utility. See “Sharing files systems in a sysplex” in *z/OS UNIX System Services Planning* for information about the z/OS UNIX support for a shared file system. See Chapter 3, “Planning the couple data sets,” on page 27 for considerations that must be reviewed when formatting a couple data set and [“z/OS UNIX parameters for format utility”](#) on page 328 for the syntax used to format the OMVS couple data set.

Using signaling for sysplex communications

Signaling is the mechanism through which XCF group members communicate in a sysplex. In a multisystem sysplex, signaling can be achieved with:

- Channel-to channel (CTC) communication connections (which can be an ESCON or FICON channel operating in CTC mode or a channel defined through a 3088 Multisystem Channel Communication Unit)
- A coupling facility (using coupling facility list structures)
- A combination of CTC connections and list structures

The implementation of signaling is transparent to exploiters of the signaling service.

Full signaling connectivity is required between all systems in a sysplex. That is, there must be an outbound and an inbound path between each pair of systems in the sysplex.

To avoid a single point of failure in the sysplex, IBM recommends that you also establish a redundant path for each required path.

Signaling services are described in [Chapter 5, “Planning signaling services in a sysplex,”](#) on page 99.

Tuning the signaling services and coupling facility

Two important areas to consider when evaluating the performance of a sysplex are the performance of the signaling service and the performance of the coupling facility.

Two RMF reports provide information that is related to the performance of a sysplex. The XCF Activity Report is a three part report that describes messages that are sent to and from a system, by:

- System
- XCF group and member
- Path

The Coupling Facility Activity Report provides information about the usage of a coupling facility, structures within a coupling facility, and subchannel connections to a coupling facility in a sysplex.

The performance of products and functions like GRS Star in a Parallel Sysplex environment depend on the coupling facility being configured to provide high-performance throughput and good service time. This cannot be done adequately in a Parallel Sysplex with a single CP to be shared between an MVS image and a coupling facility. Other types of configuration for the product or function, like GRS Ring, need to be considered. Reference Washington Systems Center Flash 10169 for further information.

Chapter 7, “Tuning a sysplex,” on page 149 briefly describes the Coupling Facility Activity Report. For more information on how to use the report, see the RMF library. Chapter 7, “Tuning a sysplex,” on page 149 also describes the factors that influence signaling performance in a sysplex and discusses how to use the XCF Activity Report to identify and analyze signaling-related constraints, performance problems, and capacity requirements.

In general, it's a good idea to get the sysplex running by using the IBM default values for parameters related to signaling. Because the signaling service attempts to optimize its use of resources, you might not need to take any additional action to ensure good signaling performance. Enabling the XCF function switch XTCSIZE on every system in the sysplex reduces the amount of tuning that might be needed. The XTCSIZE function, which is supported as of z/OS 2.4, is enabled by default.

Once the sysplex is running, you can run RMF reports to investigate how signaling resources (signaling paths and message buffers) can be tuned to improve signaling performance. Then, you can adjust resources on each system to accommodate its message traffic.

For more information about using RMF to help analyze system performance, see the RMF library.

Installing systems in a sysplex

Information on installing systems in a sysplex is included in Chapter 9, “Adding MVS systems to a sysplex,” on page 201. This chapter describes how you might build various sysplex configurations and includes diagrams for each.

Sharing a time source among LPARs

Certain processor types support the definition of a logical time offset relative to the Sysplex Timer for a particular LPAR. In a sysplex environment this function allows multiple sysplexes with different time settings to run on the same CECs with the same Sysplex Timer.

The time-of-day for each LPAR is the UTC time adjusted by the logical offset value. In a multisystem environment, XCF will allow only images with the same logical offset value to participate in the same sysplex with one another. At IPL-time, XCF detects any inconsistent logical offset values and prevents the IPL-ing system from joining the sysplex.

Chapter 2. Planning parmlib members for a sysplex

Parmlib members contain parameter values that MVS uses as input during system initialization to define the characteristics of the system.

For an MVS system to run in a sysplex, you must specify certain values in the IEASYSxx and the COUPLExx parmlib members. Certain systems running in a PR/SM partition can specify values for the XCFPOLxx parmlib member.

And, depending on the way you set up your sysplex, you can also specify values for the IEASYMxx and the LOADxx parmlib members.

This section provides background information and considerations for selecting values for these parmlib members. You might find it helpful to supplement the planning information here with the corresponding reference information from *z/OS MVS Initialization and Tuning Reference*.

Planning the IEASYSxx member in parmlib

IEASYSxx holds parameter values that control the initialization of an MVS system. Values for IEASYSxx can be part of the system parameter list or can be specified in response to the IEA101A Specify System Parameters prompt. This topic contains planning information for IEASYSxx system parameters.

Considerations for the SYSNAME system parameter

You can specify the system name on the SYSNAME parameter in the IEASYSxx parmlib member. That name can be overridden by a SYSNAME value specified either in the IEASYMxx parmlib member or in response to the IEA101A Specify System Parameters prompt.

Planning the PLEXCFG system parameter

The PLEXCFG system parameter restricts the type of sysplex configuration into which the system is allowed to IPL. For example, if you specify a multisystem sysplex (PLEXCFG=MULTISYSTEM), then you cannot specify GRS=NONE.

For System Logger setup for the PLEXCFG parameter, see [“Sysplex requirement” on page 245](#).

On PLEXCFG, specify one or more of the following:

PLEXCFG=MULTISYSTEM

Indicates that the system is to be part of a sysplex consisting of one or more MVS systems that reside on one or more processors. The same sysplex couple data sets must be used by all systems.

You must specify a COUPLExx parmlib member that identifies the same sysplex couple data sets for all systems in the sysplex (on the COUPLE statement) and signaling paths, if applicable, between systems (on the PATHIN and PATHOUT statements). You must also specify, in the CLOCKxx parmlib member, the time synchronization using ETRMODE, STPMODE, or SIMETRID.

Use MULTISYSTEM when you plan to IPL two or more MVS systems into a multisystem sysplex and exploit full XCF coupling services. GRS=NONE is not valid with PLEXCFG=MULTISYSTEM.

PLEXCFG=XCFLOCAL

Indicates that the system is to be a single, standalone MVS system that is not a member of a sysplex and cannot use couple data sets. The COUPLExx parmlib member cannot specify a sysplex couple data set, and, therefore, other couple data sets cannot be used. Thus, functions, such as WLM, that require a couple data set are not available.

In XCF-local mode, XCF does not provide signaling services between MVS systems. However, multisystem applications can create groups and members, and messages can flow between group members on this system. If signaling paths are specified, they are tested for their operational ability, but they are not used.

Use XCF-local mode for a system that is independent of other systems. In XCF-local mode, XCF services (except permanent status recording) are available on the system and you can do maintenance, such as formatting a couple data set or changing the COUPLExx parmlib member. The default COUPLE00 parmlib member is intended to be used to IPL the system in XCF-local mode, as is the specification of COUPLE=**.

PLEXCFG=MONOPLEX

Indicates that the system is to be a single-system sysplex that must use a sysplex couple data set. Additional couple data sets, such as those that contain policy information, can also be used. XCF coupling services are available on the system, and multisystem applications can create groups and members. Messages can flow between members on this system (but not between this system and other MVS systems) via XCF signaling services. If signaling paths are specified, they are not used.

You must specify a COUPLExx parmlib member that gives the system access to a sysplex couple data set to be used only by this system. When a system IPLs into a single-system sysplex, no other system is allowed to join the sysplex.

Use MONOPLEX when you want only one system in the sysplex (for example, to test multisystem applications on one system) or when you want to use a function, such as WLM, that require a couple data set.

PLEXCFG=ANY

Indicates that the system can be part of any valid sysplex configuration. Specifying ANY is logically equivalent to specifying XCFLOCAL, MONOPLEX, or MULTISYSTEM. ANY is the default. If the system is ever to be part of a single-system sysplex or a multisystem sysplex, you must specify a COUPLExx parmlib member that gives the system access to a couple dataset. Note that the default COUPLE00 parmlib member does not include a PCOUPLE statement, and therefore is intended to be used only to bring up the system in XCF-local mode. Specifying COUPLE=** will also allow the system to be brought up in XCF-local mode. See the COUPLE parameter of IEASYSxx in *z/OS MVS Initialization and Tuning Reference* additional information about COUPLE=**.

When PLEXCFG=ANY is specified, MVS will initialize and establish a PLEXCFG “mode” of XCFLOCAL, MONOPLEX, or MULTISYSTEM based on parmlib settings in COUPLExx and CLOCKxx as well as the availability of sysplex resources. For example:

- If you specify PLEXCFG=ANY and a sysplex couple data set is not specified in COUPLExx, or if COUPLE=** is specified, the MVS system will be initialized in XCF-local mode.
- If you specify PLEXCFG=ANY and a sysplex couple data set is specified in COUPLExx but the data set is not available, the MVS system can be initialized in XCF-local mode either by switching to a COUPLExx member that does not specify sysplex couple data sets or by specifying COUPLE=**.
- If you specify PLEXCFG=ANY and a sysplex couple data set is specified and available via COUPLExx, and a sysplex timer is specified (ETRMODE YES in CLOCKxx) but ETR signals are not present, the MVS system can be initialized in MONOPLEX mode. The system is also considered to be in ETR-LOCAL mode. If ETR signals are restored and the CPC TOD and the ETR TOD are resynchronized (thus establishing ETR mode), the MVS system will be in a sysplex that is MULTISYSTEM-capable.
- If you specify PLEXCFG=ANY and a sysplex couple data set is specified and available via COUPLExx and a sysplex time reference is specified (CLOCKxx specifies either SIMETRID or ETRMODE YES) and available, the MVS system will be in a system that is MULTISYSTEM-capable.

PLEXCFG=ANY is the least restrictive of the PLEXCFG parameters and can provide flexibility during system initialization that can be useful when a sysplex couple data set or Sysplex Timer are not available. The first goal, however, should be to provide an adequate level of resource redundancy to help prevent availability problems, and then specify the desired configuration for the PLEXCFG parameter.

When using PLEXCFG=ANY, ensure that systems requiring a multisystem configuration do not run concurrently with each other in XCF-local mode. For example, if system “A” and system “B” normally run in a multisystem sysplex using global resource serialization to serialize access to a shared common data base, you would not want to bring up “A” and “B” in XCF-local mode using two different global resource serialization complexes. To do so would introduce data integrity exposures to the shared common data base.

Programming note: To prevent the operator from overriding the PLEXCFG parameter, use OPI=NO on the PLEXCFG keyword in the IEASYSxx parmlib member. (For example, PLEXCFG=MULTISYSTEM,OPI=NO). Note, however, that IBM recommends that OPI=YES be used in case PLEXCFG=XCFLOCAL needs to be specified to IPL the system in XCF-local mode to format the sysplex couple data set.

Planning for XCF-local mode

In certain recovery scenarios, it might be necessary to have a system that can be initialized in XCF-local (standalone system) mode. To IPL a system in XCF-local mode, do the following:

1. Define a separate set of parmlib members to be used to IPL the systems in XCF-local mode.
 - Define an alternate LOADxx parameter that points to an IEASYSxx parmlib member that will be used to initialize a system in XCF-local mode.
 - Define an IEASYSxx parmlib member that will be used to initialize a system in XCF-local mode. Some of the parameters that might need to be specified are:
 - PLEXCFG=XCFLOCAL
 - GRS=NONE (or GRS=START or GRS=JOIN when global resource serialization CTCs are defined)
 - COUPLExx (or COUPLE=**)

You may choose to alter other parameters or parmlib members depending on your hardware and software configuration.
2. If COUPLE=** is not used, define a COUPLExx parmlib member with one of the following specified:
 - COUPLE SYSPLEX(&SYSPLEX) — If the sysplex name is contained in LOADxx.
 - COUPLE SYSPLEX(LOCAL) — If the sysplex name is not contained in LOADxx.

(Do not specify any other statements in COUPLExx.)
3. Create a separate set of system packs for the XCF-local system.
4. Create a set of IODFs that:
 - a. Prevents access to data that is in use by other systems (used to test the basic function of the XCF-local system).
 - b. Permits access to data that is used by other systems (used to access shared data when all other systems are down).
5. Create a set of operational procedures necessary to initialize and use the XCF-local system.

To preserve flexibility of configuration options during IPL, specify OPI=YES to allow the operator to change the configuration if necessary. See the OPI parameter of IEASYSxx in [z/OS MVS Initialization and Tuning Reference](#) for additional information.

Take the necessary precautions to ensure that the XCF-local system does not access data concurrent with any other systems because the XCF-local system may not be serializing shared data. For example, if your sysplex processes a common workload and accesses shared data bases using mechanisms like global resource serialization to ensure serialized access to shared data, an XCF-local system should not be allowed to access the same shared data concurrent with your sysplex because the XCF-local system exists outside your sysplex and may not provide serialized access to shared data.

Planning the GRS system parameter

The GRS system parameter indicates whether the system is to join a global resource serialization complex.

In a multisystem sysplex, every system in the sysplex must be in the same global resource serialization complex. This allows global serialization of resources in the sysplex.

To initialize each MVS system in the multisystem sysplex, you must use the global resource serialization component of MVS. You can use another product to serialize the use of shared resources, but global resource serialization is also required.

System Logger has special requirements for multiple sysplexes in a GRS Ring Complex environment. See [“Sysplex requirement” on page 245](#).

For a global resource serialization star complex, specify

GRS=STAR

The effect that STAR has depends on the system configuration as follows:

XCF-local mode (PLEXCFG=XCFLOCAL)

This configuration is not allowed in a star complex. Message ISG303D is issued, and the operator is prompted to either re-IPL the system or to continue the IPL with GRS=NONE.

Single-system sysplex (PLEXCFG=MONOPLEX)

This configuration is not allowed in a star complex. Message ISG303D is issued, and the operator is prompted to either re-IPL the system or to continue the IPL with GRS=NONE.

Default sysplex configuration (PLEXCFG=ANY)

If XCF is in local mode, this configuration is not allowed. Otherwise, the system is initialized into a global resource serialization star complex and XCF protocols are used in the complex.

Multisystem sysplex (PLEXCFG=MULTISYSTEM)

The system starts or joins an existing sysplex and a global resource serialization star complex. XCF coupling services are used in the sysplex and in the complex.

For a global resource serialization ring complex, specify one of the following:

GRS=JOIN or START or TRYJOIN

The effect of JOIN, START, or TRYJOIN depends on the sysplex configuration, as follows:

XCF-local mode (PLEXCFG=XCFLOCAL)

There is only one system (in XCF-local mode) in the sysplex. The system starts or joins an existing global resource serialization complex and non-XCF protocols are used in the complex.

Single-system sysplex (PLEXCFG=MONOPLEX)

There is only one system in the sysplex. The system starts or joins an existing global resource serialization complex and non-XCF protocols are used in the complex.

Multisystem sysplex (PLEXCFG=MULTISYSTEM)

The system starts or joins an existing sysplex and a global resource serialization complex. XCF coupling services are used in the sysplex and the complex. (For systems in the complex that are not in the sysplex, non-XCF protocols are used.)

GRS=NONE

The effect of NONE depends on the intended sysplex configuration, as follows:

XCF-local mode (PLEXCFG=XCFLOCAL)

There is only one system (in XCF-local mode) in the sysplex and no global resource serialization complex.

Single-system sysplex (PLEXCFG=MONOPLEX)

There is only one system in the sysplex and no global resource serialization complex.

Multisystem sysplex (PLEXCFG=MULTISYSTEM)

This is not allowed. Global resource serialization is required in a multisystem sysplex.

Summary of PLEXCFG and GRS system parameters

If you do not have an existing global resource serialization complex:

- Use GRS=NONE and PLEXCFG=XCFLOCAL or MONOPLEX when you want a single-system sysplex and do not want a global resource serialization complex.
- Use GRS=TRYJOIN (or GRS=START or JOIN) and PLEXCFG=MULTISYSTEM when you want a sysplex of two or more systems and want a global resource serialization ring complex that matches the sysplex and uses XCF services.
- Use GRS=STAR and PLEXCFG=MULTISYSTEM when you want a global resource serialization star complex.

If you have an existing global resource serialization complex:

- Use GRS=START or JOIN and PLEXCFG=XCFLOCAL or MONOPLEX when you want a single-system sysplex and want to maintain an existing global resource serialization complex that uses non-XCF protocols.
- Use GRS=START or JOIN and PLEXCFG=MULTISYSTEM when you want a sysplex of two or more systems and want to migrate systems in an existing global resource serialization complex. Systems that are in both the sysplex and complex use XCF services; systems that are not in the sysplex but are in the complex, use non-XCF protocols.

See *z/OS MVS Planning: Global Resource Serialization* for additional planning information for the GRS system parameter. See [“Multisystem sysplex configuration without an existing global resource serialization complex”](#) on page 203 and [“Multisystem sysplex configuration with an global resource serialization complex”](#) on page 204 for installing information.

Planning the GRSCNF system parameter

The GRSCNF=xx system parameter specifies the GRSCNFxx parmlib member that describes the global resource serialization complex for the initializing system.

Use GRSCNF=00 (the default) when all systems in the sysplex are in the global resource serialization ring complex and you can use the values in the default GRSCNF00 parmlib member to provide control information about the complex. With GRSCNF00, all global resource serialization communication is through XCF services.

Note: If GRSCNF00 is used to initialize a star complex, only the MATCHSYS and CTRACE parameters have meaning. The presence of other ring-related parameters is “tolerated”. An informational message is issued and the parameters are ignored other than for syntax checking.

Use GRSCNF=xx when you have a global resource serialization mixed complex (one or more systems in the complex are not part of a sysplex) and need to provide control and configuration information consistent with the complex. There is no requirement for a GRSCNFxx parmlib member to initialize a star complex, unless you are using a CTRACE parmlib member other than the default, CTIGRS00, shipped by IBM.

See *z/OS MVS Planning: Global Resource Serialization* for planning information for the GRSCNF system parameter and the GRSCNF parmlib member.

Planning the GRSRNL system parameter

The GRSRNL=00 system parameter specifies the GRSRNL00 parmlib member.

There are several ways to specify the GRSRNL system parameter; GRSRNL=00, GRSRNL=(xx,yy...) or GRSRNL=EXCLUDE.

Use the GRSRNL=00 system parameter when you can use the default RNLs in the GRSRNL00 parmlib member for global resource serialization in the complex.

The GRSRNL=(xx,yy...) system parameter specifies the GRSRNLxx parmlib member(s) that consist of the resource name lists (RNLs) used in the complex to control the serialization of resources.

Use GRSRNL=(xx,yy...) when you have an existing global resource serialization complex, are adding the system to a sysplex, and want to use the existing RNLs in the complex.

GRSRNL=EXCLUDE indicates that all global enqueues are to be treated as local enqueues. All ENQ macro requests are excluded from global serialization and are serialized locally; except that all resources identified on an ENQ SCOPE=SYSTEMS,RNL=NO macro are globally serialized on all systems in the complex.

Use GRSRNL=EXCLUDE when you do not have an existing global resource serialization complex, are adding the system to a sysplex, and want the serialization of most resources done with the RESERVE macro.

See *z/OS MVS Planning: Global Resource Serialization* for planning information for the GRSRNL system parameter and the GRSRNL parmlib member.

Planning the CLOCK system parameter

The CLOCK=xx system parameter specifies the CLOCKxx parmlib member to be used when you IPL your system. CLOCKxx indicates how the time of day (TOD) is to be set on the system.

Setting the TOD clock

MVS bases its decisions and activities on the assumption that time is progressing forward at a constant rate. The instrument used to mark time in an MVS system is the time of day (TOD) clock. As it operates, MVS obtains time stamps from the TOD clock. MVS uses these time stamps to:

- Identify the sequence of events in the system
- Determine the duration of an activity
- Record the time on online reports or printed output
- Record the time in online logs used for recovery purposes.

To ensure that MVS makes time-related decisions as you intend, IBM recommends that you **do not reset your TOD clock**, for example, to switch to or from Daylight Savings Time. Instead, set the TOD clock to a standard time origin, such as Greenwich Mean Time (GMT), and use the TIMEZONE statement of the CLOCKxx parmlib member to adjust for local time. (The TIMEZONE statement allows you to specify how many hours east or west of Greenwich you are.) To adjust local time, change the offset value for the TIMEZONE statement. This will not disturb MVS's assumption that time is progressing forward and will allow time stamps on printed output and displays to match local time.

Synchronizing time stamps in a sysplex

Similarly, for MVS to monitor and sequence events in a sysplex, time must be synchronized across all systems. Thus, MVS requires that each processor in the sysplex share a common time source that can provide synchronized time stamps. An IBM Sysplex Timer (9037) traditionally has been used for this purpose. On processors with the Server Time Protocol (STP) architecture, a Coordinated Timing Network (CTN) can also be used by multiple servers to maintain time synchronization.

Starting with the z114/z196 servers, STP is required; it has replaced the Sysplex Timer and is fully compatible with the Sysplex Timer. This compatibility allows for several migration path variations. Thus, it is possible to go from a Sysplex Timer-only mode where all processors are kept in synchronization through direct connections to the Sysplex Timer, to mixed mode, where STP and the Sysplex Timer work together to keep processors in synchronization, to full STP-mode, where the processors are kept in synchronization through STP only.

STP provides time synchronization by allowing multiple servers to form a CTN over data links (ISC-3 and IFB) between servers. A CTN identifier (CTN ID) identifies the CTN, which can be defined as either an STP-only network or a mixed network consisting of both a CTN network and a Sysplex Timer network. Configuration information for each server identifies the CTN ID as the timing network for the server.

Adjusting local time in a sysplex

The following actions are important whether or not your installation has decided to operate using the TOD clock set to the local time of the installation, or GMT time. It is of the utmost importance that the TOD clock value not be changed on any of the processors in the sysplex once a value has been established for a running sysplex. The TOD clock on all processors must be using the identical time, or be synchronized to the same time, because that is the value used by software. The value known as local time, which is computed as an offset from the TOD clock value, may be changed at any time. It is not the value used by time-critical software to perform tasks. To adjust local time without resetting the TOD clock in the processor, you can:

- Reset the time offset in the Sysplex Timer

- Change the local time offset in the CLOCKxx member of parmllib
- Issue the SET CLOCK command
- Adjust time zone settings with STP, using the HMC

The method you choose to adjust local time depends on your sysplex configuration and on the options specified on the ETRMODE, ETRZONE, and SIMETRID statements of the active CLOCKxx member.

1. Sysplex includes a Sysplex Timer

- If the time-zone offset is taken from the Sysplex Timer (ETRMODE YES and ETRZONE YES), change the time offset from the Sysplex Timer console. See *Planning for the 9037 Sysplex Timer* (GA23-0365) for additional details.

There is no need to reIPL the system to cause the local time adjustment to take effect.

- If the time-zone offset is taken from the CLOCKxx member (ETRMODE YES and ETRZONE NO) do either of the following:
 - Issue the SET CLOCK command to change the local time offset and change the local time offset value in the CLOCKxx member to reflect the new offset. This will preserve the change for ensuing IPLs.
 - Change the offset value in CLOCKxx and re-IPL.

2. Sysplex does not include a Sysplex Timer

- In a single system sysplex, do either of the following:
 - Issue the SET CLOCK command to change the local time offset, and change the local time offset value in the CLOCKxx member to reflect the new offset. This will preserve the change for ensuing IPLs.
 - Change the offset value in CLOCKxx and re-IPL.
- In a multisystem sysplex on one processor (either guest MVS systems running under a host VM system or in logical partitions (LPARs) in a PR/SM environment), do either of the following:
 - Update the CLOCKxx member for each guest MVS system with the correct time-zone value and reIPL each system.
 - Issue the SET CLOCK command to change the local time.

Systems that are not attached to the Sysplex Timer must be reIPLed in order to begin using the new time-zone value in CLOCKxx.

For more information about STPMODE settings, see [Server Time Protocol Planning Guide \(www.redbooks.ibm.com/redbooks/pdfs/sg247280.pdf\)](http://www.redbooks.ibm.com/redbooks/pdfs/sg247280.pdf).

Understanding PLEXCFG configurations with a Sysplex Timer

The ETRMODE parameter in your CLOCKxx member indicates whether you are using a Sysplex Timer to synchronize time. Specify ETRMODE YES if you want MVS to use the Sysplex Timer. The STPMODE parameter in your CLOCKxx member indicates whether MVS can use STP timing mode for time synchronization. Specify STPMODE YES if you want MVS to use STP timing mode.

- In a non-sysplex (PLEXCFG=XCFLOCAL) or single-system sysplex (PLEXCFG=MONOPLEX), the use of external time synchronization is optional.
- In a multisystem sysplex (PLEXCFG=MULTISYSTEM), external time synchronization is required if the systems in the sysplex span hardware CPCs. If all systems in the sysplex are contained within one physical partition of a single CPC, you can choose either to use external time synchronization (ETRMODE YES, or STPMODE YES, or both), or use the local TOD clock to synchronize the MVS images (SIMETRID (nn)).

A Sysplex Timer failure can affect the sysplex in several ways, depending on how the sysplex is configured with the PLEXCFG system parameter. In general, when an MVS image is running and loses Sysplex Timer signals:

- If the MVS image is in a configuration that is multisystem sysplex capable, message IEA015A is issued requiring a reply of either RETRY or ABORT. If ABORT is replied to IEA015A, the MVS image is placed into a non-restartable wait state.

Exception: If the GDPS/PPRC controlling system loses time synchronization in a multisystem-capable sysplex, the system is not immediately placed into the wait state, nor is message IEA015A issued for such a system. It is allowed to remain running for a limited amount of time to guarantee a consistent set of secondary PPRC volumes.

- If the MVS image is in a configuration that is non-sysplex or single system sysplex mode, the MVS image is placed in local timer mode. This mode will not affect the operation of the MVS image in the configuration.

To avoid outages due to lost Sysplex Timer signals, you should configure redundant Sysplex Timers and Sysplex Timer links, and also eliminate single points of failure such as a single power source.

The following information summarizes the effects of a Sysplex Timer failure for each of the PLEXCFG configuration options. Two scenarios are presented: the first illustrates the effect when Sysplex Timer signals are not present at IPL; the second illustrates the effect when Sysplex Timer signals are lost while MVS is running.

- **XCF-local mode (PLEXCFG=XCFLOCAL)**

- If no Sysplex Timer signals are available at IPL, the system is brought up in local timer mode, using the system TOD clock.
- If Sysplex Timer signals are lost, the system enters local timer mode.

- **Single-system sysplex (PLEXCFG=MONOPLEX)**

- If no Sysplex Timer signals are available at IPL, the system is brought up in local timer mode, using the system TOD clock.
- If Sysplex Timer signals are lost, the system enters local timer mode.

- **Multisystem sysplex (PLEXCFG=MULTISYSTEM)**

- If no Sysplex Timer signals are available at IPL, MVS initialization cannot complete.
- If Sysplex Timer signals are lost, message IEA015A is issued. If the reply is ABORT, the affected MVS image is placed in a non-restartable wait state (WAIT0A2).

Exception: A GDPS® controlling system is allowed to remain running for a limited time to guarantee a consistent set of secondary PPRC volumes.

- **Default sysplex configuration (PLEXCFG=ANY)**

- If a sysplex couple data set is not specified or not available via COUPLExx, the implied configuration is XCF-local mode:
 - If no Sysplex Timer signal is available at IPL, the system is brought up in local timer mode, using the system TOD clock.
 - If a Sysplex Timer signal is lost, message IEA015A is issued. If the reply is ABORT, the system enters local timer mode.
- If a sysplex couple data set is specified and available via COUPLExx, the implied configuration is MONOPLEX or MULTISYSTEM depending on the availability of Sysplex Timer signals:
 - During the IPL of the FIRST system, if Sysplex Timer signals are not available, the system is brought up in local timer mode using the system TOD clock. Additional systems cannot be brought into the sysplex until Sysplex Timer signals are restored.
 - If Sysplex Timer signals are restored, and the CPC TOD and the Sysplex Timer TOD are resynchronized, the sysplex becomes MULTISYSTEM-capable and will allow other systems to join the sysplex.
 - If Sysplex Timer signals are lost, message IEA015A is issued. If the reply is ABORT, the affected MVS image is placed in a non-restartable wait state (WAIT0A2).

Exception: A GDPS controlling system is allowed to remain running for a limited time to guarantee a consistent set of secondary PPRC volumes.

Note: When MVS is running in local timer mode, if Sysplex Timer signals are restored, the affected MVS image will enter Sysplex Timer mode if the CPC TOD and the Sysplex Timer TOD become resynchronized. Resynchronization of CPC and Sysplex Timer TODs can occur when the difference between the CPC TOD and the Sysplex Timer TOD does not exceed the value of ETRDELTA specified in CLOCKxx.

Consequences of resetting the TOD clock in a sysplex

Never change the TOD clock that the sysplex is using, once it is set. Doing so can lead to a sysplex-wide IPL, or having working systems within the sysplex at risk of being partitioned out of the sysplex (that is, placed in a non-restartable wait state). Whether you choose to set the TOD clock to local time or to GMT time, local time changes can be accomplished using one of the techniques discussed in [“Adjusting local time in a sysplex” on page 14](#). You should be aware of the possible consequences to XCF and the multisystem applications using XCF services when you set the TOD clock backward or forward:

- **Setting the TOD Clock Backward**

As it monitors system status, MVS records, in the sysplex couple data set, the time stamps obtained from the TOD clock. If you set the TOD clock back, then time stamps obtained from the clock may duplicate time stamps already recorded in the sysplex couple data set. For example, if the TOD clock is reset to 2:00 AM at 3:00 AM, then all timings between 2:00 and 3:00 will occur again. When a system attempts to IPL into the sysplex, MVS compares time stamps from the TOD clock with time stamps recorded in the sysplex couple data set. If time stamps in the couple data set are later than those specified by the incoming system, MVS determines this is an error and does not allow the system to join the sysplex. The system is not allowed to IPL into the sysplex until the time stamp from the TOD clock is greater than time stamps recorded in the sysplex couple data set. In the example described, this condition would persist for one hour.

Furthermore, it is common practice for many MVS applications to record time stamps on external media. For example, time stamps might be written to a data set containing a database log. These logs could be used by several processes, such as database recovery, which depend critically on progressively increasing time stamps to preserve the ordering of events. Setting the TOD clock backward could make the log useless for database recovery.

- **Setting the TOD Clock Forward**

As part of its status monitoring, MVS periodically checks the status of each system in the sysplex. To determine system failure, MVS uses the failure detection interval value specified in the COUPLExx parmli member. If you set the TOD clock ahead, then the difference between a time stamp previously recorded in the couple data set and a time stamp obtained from the reset TOD clock could exceed the specified failure detection interval. In this case, MVS would initiate a “status update missing” condition, which in turn would cause the other system(s) to be removed from the sysplex (that is, placed in a non-restartable wait state). This would leave only the system with the future TOD clock in the system or sysplex.

Planning the COUPLE system parameter

The COUPLE=xx system parameter specifies the COUPLExx parmli member used when you IPL your system. See [“Planning the COUPLExx member in parmli” on page 18](#) for planning information for this member.

If you do not specify the COUPLE=xx parameter, COUPLE=00, which specifies the COUPLE00 parmli member, is used. The IBM-supplied default COUPLE00 parmli member causes the system to come up in single system sysplex mode without couple data sets, (which is also called XCF-local mode).

LOCAL is the sysplex name in the IBM-supplied default COUPLE00 parmli member. The name LOCAL has no special meaning to XCF. In XCF-local mode, the installation can specify any sysplex name. Note, however, that the IPL will fail when using the IBM-supplied COUPLE00 parmli member if a sysplex name other than LOCAL is specified in LOADxx.

Specifying COUPLE=**

COUPLE** is not an actual parmlib member, but a specification of COUPLE=** is understood by XCF to indicate that the system is to be IPLed in XCF-local mode. This is particularly useful when the system must be IPLed in order to rectify parmlib errors in COUPLExx that otherwise cause the IPL to fail.

Like the default COUPLE00 parmlib member, COUPLE=** uses a sysplex name of LOCAL, except when a sysplex name other than LOCAL is specified in LOADxx. In that case, COUPLE=** will use the name specified in LOADxx, thus allowing the IPL to proceed.

Planning the IXGCNF system parameter

The IXGCNF=system parameter specifies one or more IXGCNFxx parmlib members to be used when system logger starts or is restarted on the initializing system within the sysplex.

The IXGCNFxx parmlib members contain statements that can be used to control the following functions:

1. Identify the tracing options when system logger initializes.
2. Specify the logger monitoring intervals for warning and action messages.
3. Specify the logger resource management policies on a system.
4. Specify the ZAI statement attributes for the z/OS Advanced Workload Analysis Reporter (IBM zAware) log stream client socket communication and log data buffering details in order to send log stream data to the IBM zAware server.

Syntax IXGCNF=aa specifies a single member and syntax IXGCNF=(aa,bb,...) identifies groups of system logger initialization statements across several IXGCNFxx members. When multiple IXGCNFxx members are concatenated, the individual system logger options are merged with the last parmlib member option taking precedence.

If any errors are encountered for the specified IXGCNFxx parmlib members, logger will issue an error message and then attempt to process remaining parameters for syntax. For a SET IXGCNF command, parameter updates are not done. At IPL, default values are substituted for all parameters. For example, assume a shared IXGCNFxx parmlib includes the MONITOR LSPRIMARY,CONSUMPTIONALERT(value) specification. Also assume one release is at z/OS v1r13 and the other is at z/OS v2r1. The shared IXGCNFxx parmlib member will be honored on the z/OS v2r1 release. However, the z/OS v1r13 release processing will issue an unrecognized keyword type error message, and use the system defaults for all the IXGCNFxx options.

The DISPLAY LOGGER,IXGCNF[,options] command displays the merged information.

See [“Offloading log data from interim storage by freeing and/or moving it to DASD” on page 222](#) and [“Offload and service task monitoring” on page 227](#) for more information about system logger log stream offload processing and monitoring activities.

See [“Prevent a z/OS image from accessing sysplex scope LOGR couple data sets” on page 277](#) for more information about LOGR CDS management.

See [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex” on page 279](#) for more information about using a system logger couple data set for just one system.

See [“Preparing for z/OS IBM zAware log stream client usage” on page 311](#) for information on designing which z/OS image will make use of the IBM zAware server capabilities.

Planning the DRMODE System Parameter

The DRMODE system parameter is no longer used by system logger. Any specification of it, has no effect on logstream recovery processing.

Planning the COUPLExx member in parmlib

The COUPLExx parmlib member contains values that reflect fundamental decisions an installation makes as it sets up a sysplex.

The COUPLE statement

The COUPLE statement includes the following keywords:

- **SYSplex** — contains the name of the sysplex. The sysplex name allows a system to become part of the named sysplex.

Specify the same sysplex-name for each MVS system in the sysplex. Sysplex-name must match the sysplex name specified on the couple data sets when they were formatted.

If you plan to define two or more sysplexes, IBM recommends that their names be unique. Some applications cannot handle duplicate sysplex names. Also, consider using meaningful names. For example, if you plan to set up three sysplexes in a New York installation, two in San Francisco, and another in Dallas, the sysplex names could be defined as: NYPLEX00, NYPLEX01, NYPLEX02, SFPLEX00, SFPLEX01, and DAPLEX00.

The sysplex name is also the substitution text for the &SYSplex system symbol. As of MVS/ESA SP 5.2, if you specify the sysplex name in both the COUPLExx and LOADxx parmlib members, all parmlib members can use the defined substitution text for &SYSplex.

- **PCOUPLE, ACOUPLE** — contain the names of the primary and alternate sysplex couple data sets and are described in [Chapter 3, “Planning the couple data sets,”](#) on page 27.
- **INTERVAL, OPNOTIFY, CLEANUP** — can contain values that reflect recovery-related decisions for the sysplex. They are described in the topic, [“Controlling availability and recovery through COUPLExx”](#) on page 178.
- **CTRACE** — can name an XES or XCF component trace parmlib member that contains initialization parameters for its component trace.

In the CTRACE keyword, you can specify a CTnXCFxx, a CTnXESxx parmlib member, or both (by specifying the CTRACE keyword twice). For information about component traces, see [z/OS MVS Diagnosis: Tools and Service Aids](#).

- **MAXMSG, RETRY, CLASSLEN** — can contain values related to signaling services and are discussed in [“Planning for optimum signaling performance”](#) on page 105. A value you specify for one of these keywords on the COUPLE statement becomes the default value if you do not specify that keyword on the CLASSDEF, PATHIN, PATHOUT, or LOCALMSG statement.
- **CFRMPOL** — can contain the name of a CFRM policy that is to be automatically started at IPL time, if there is no other previously-activated CFRM policy in effect. See [“Starting a CFRM policy”](#) on page 66.
- **CFRMOWNEDCFPROMPT** — allows the installation to request operator prompting to control usage of coupling facilities previously owned by the same sysplex at IPL time.
- **VMCPUIDTOLERATION** — controls the verification of CPU identification information.

The FUNCTIONS statement

The FUNCTIONS statement describes the initial enablement state of optional functions that are provided by the XCF and XES components of z/OS. The controllable functions are described in [Table 1](#) on page 19. The names that are listed in the table are the function names that are used in the COUPLExx FUNCTIONS statement (see [z/OS MVS Initialization and Tuning Reference](#) for the syntax) and the SETXCF FUNCTIONS command (see [z/OS MVS System Commands](#) for the syntax).

Table 1. XCF/XES Optional Functions		
Name	Function	Default State
DUPLEXCF16	Enhancements to the system-managed duplexing protocol, introduced with CFLEVEL 16. When this protocol is enabled, certain duplexed commands can expedite the exchange of ready-to-complete signals between coupling facilities.	Disabled

Table 1. XCF/XES Optional Functions (continued)		
Name	Function	Default State
CFLCRMGMT	Enhancements to the CFRM message-based protocol introduced with z/OS V2R1. When this function is enabled on all active systems in the sysplex, the message-based manager system can enable CFRM LOSSCONN recovery management. CFRM LOSSCONN recovery management can enhance average CF LOSSCONN recovery time by processing CF structures serially, rather than in parallel.	Disabled
SYSSTATDETECT	Enhancements to the partitioning protocol for removing a system from the sysplex. When this function is enabled, the partition monitoring system detects changes in the target system status, and might be able to bypass certain delays built into the partitioning process. This support also enables the system to initiate actions, such as reset-normal, against the remote system to ensure that it is no longer actively processing.	Enabled
USERINTERVAL	<p>When this function is enabled, the effective failure detection interval is set to the last explicitly specified INTERVAL value from one of the following sources:</p> <ul style="list-style-type: none"> • The INTERVAL parameter on the COUPLExx parmlib member • The SETXCF COUPLE,INTERVAL command • The cluster instrumentation software <p>If the INTERVAL has not been explicitly specified, the effective failure detection interval is set to be the default value derived from the excessive spin parameters.</p>	Disabled
CRITICALPAGING	<p>This function controls the RSM page fault hardening behavior. When this function is enabled, the system tries not to page out to DASD the following:</p> <ul style="list-style-type: none"> • 31-bit common storage • Address spaces that are defined as critical for paging. • All data spaces associated with those address spaces that are critical for paging. • Pageable link pack area (PLPA) <p>If the system is experiencing a critical real frame shortage, these storage areas might be paged out by the system as a last resort. To enable the CRITICALPAGING function, use the FUNCTIONS statement in the COUPLExx parmlib member. You cannot enable the CRITICALPAGING function through the SETXCF command. However, you can use either method to disable CRITICALPAGING. For information about indicating the address spaces that are critical for paging, see SCHEDxx (PPT, master trace table, and abend codes for automatic restart) in <i>z/OS MVS Initialization and Tuning Reference</i>.</p>	Disabled
DUPLEXCFDIAG	When this function is enabled and the completion of a duplexed coupling facility request is delayed, software and hardware problem determination data is to be collected. The intent of this support is to collect coherent data from z/OS, coupling facilities, and message paths when apparently a coupling facility might break duplexing for the structure that is associated with the delayed command. The function is available only when the structure is duplexed between two coupling facilities at CFLEVEL 17 or above, and when the link firmware supports the data collection function.	Disabled

Table 1. XCF/XES Optional Functions (continued)		
Name	Function	Default State
COUPLINGTHININT	When this function is enabled, coupling thin interrupts in the channel subsystem, which provide for improved performance and throughput for coupling facility workloads can be enabled in the channel subsystem (when available in the hardware) to deliver coupling-related adapter interrupts for coupling facility engines.	Enabled
CFSTRQMON	When this Coupling Facility Structure Queue Monitoring function is enabled, coupling facility (CF) requests that are delayed longer than expected trigger messages that warn of a potential problem. Refer to messages IXL053E, IXL054I, IXL055I, and IXL056I for additional information.	Disabled
MSGISO	XCF Message Isolation. When this function is disabled, the system will no longer isolate a member when it fails to make adequate progress regarding the processing of its messages. Message isolation helps keep problematic group members from impeding the delivery of messages to other members. While a member is isolated, XCF delays or rejects messages targeted to that member. Enabling message isolation enhances XCF's ability to protect the sysplex from problematic group members. Disabling message isolation can degrade signal performance because it reduces XCF's ability to overcome inbound "stall" conditions.	Enabled
XTCSIZE	The XCF function switch XTCSIZE determines how size-only transport class definitions are used. If XTCSIZE is disabled or the XTCSIZE function is not supported by the target system, you must continue to create suitable transport class definitions. Traditional transport class rules are used when sending signals. When XTCSIZE is enabled and the target system supports the XTCSIZE function, the signal resources that are assigned to the size-only transport classes become XCF Managed. When XCF Managed, signal paths are tuned for the maximum signal size and best fit buffers are used to send signals. When XTCSIZE is enabled on all systems in the sysplex, you no longer need to define, tune, or manage size-only transport classes.	Enabled
CFMONOPAVOID	<p>This function controls the z/OS response to detection of structure monopolization of coupling facility (CF) resources. When a structure is allocated in a CF at or above CFLEVEL 24, the CF indicates to z/OS when the structure appears to be consuming a disproportionate share of CF resources. When this function is enabled on a z/OS system, that z/OS instance responds to the CF notification by limiting CF requests for that structure until the monopolization condition clears. This function is intended to prevent "runaway" structure-exploiting applications from impacting other well-behaved applications.</p> <p>Enable the CFMONOPAVOID function to prevent individual structures from consuming an excessive proportion of CF resources.</p> <p>Disable the CFMONOPAVOID function if apparently the throttling actions taken by z/OS, in response to reported CF monopolization, are adversely impacting an application that is behaving as expected. For example, the CF can report monopolization for a structure that is expected to process a heavy workload and thus is expected to consume more CF resources than other structures. Throttling requests to such a structure would be undesirable.</p>	Enabled

CLASSDEF, PATHIN, PATHOUT, and LOCALMSG statements

The CLASSDEF, PATHIN, PATHOUT, and LOCALMSG statements contain values to help tailor signaling options in the sysplex. These statements are discussed in ["Planning for optimum signaling performance" on page 105](#).

The DATA statement

A DATA statement contains the name of the couple data sets that are to hold policy information for a specified policy type. The DATA statement is discussed in [Chapter 3, “Planning the couple data sets,”](#) on [page 27](#).

Modifying COUPLExx values

To change values specified on the COUPLE statement of the COUPLExx parmlib member, use the SETXCF COUPLE command.

To change the enablement state of an optional XCF/XES function, use the SETXCF FUNCTIONS command.

To start or stop an inbound or outbound signaling path or to create or delete a transport class, use the SETXCF START or SETXCF STOP command.

To modify local messages, use the SETXCF MODIFY,LOCALMSG command.

SETXCF commands do not change the values specified in the COUPLExx parmlib member. Therefore, to make permanent the changes you have made using SETXCF commands, ensure that you modify COUPLExx appropriately before the next IPL.

Planning the CONSOLxx member in parmlib

The CONSOLxx parmlib member contains values that define your console configuration. CONSOLxx is used to specify the attributes of multiple console support (MCS) and SNA multiple console support (SMCS) consoles, the types of commands operators can issue, and routing information for messages. IBM recommends that you use the same CONSOLxx member to describe all MCS and SMCS consoles in the sysplex.

In a sysplex, some values defined in CONSOLxx have *sysplex scope*. When a system with those values is first IPLed into a sysplex, the values are in effect for the entire sysplex. See [z/OS MVS Planning: Operations](#) for complete information about specifying CONSOLxx values in a sysplex environment.

Naming your consoles

A console name uniquely identifies a console to the sysplex for the life of the sysplex.

Understanding message routing

CONSOLxx defines console attributes to control the types of commands operators can issue and routing information for messages. The recommendations in this topic apply when specifying the CONSOLxx parmlib member in a sysplex environment.

The CONSOLE statement of CONSOLxx contains the following:

- CMDSYS(*) — To direct commands issued on the console to the system on which the console is attached. Specifying CMDSYS(*) defines command association between a console and a system in the sysplex.
- MSCOPE(*ALL) — To indicate that messages issued by all systems in the sysplex can be received by the console.
- DEVNUM(SYSCONS) — To define the system console. The system console is used as a backup for recovery purposes and can also be used to IPL a system or, if necessary, to run the sysplex.

Related parmlib members to consider

The Message Processing Facility (MPF) can control message processing. Specify the MPFLST members to use on the CONSOLxx INIT statement.

The Subsystem Definitions (IEFSSNxx) parmlib member contains parameters that define the primary and secondary subsystems in your installation, and the relevant command prefix used by the subsystem.

Planning the CLOCKxx member in parmli

The CLOCKxx parmli member contains values that do the following for each system in a sysplex:

- Specify that the system is to prompt the operator to initialize the time of day (TOD) clock during system initialization.
- Specify the difference between the local time and Greenwich Mean Time (GMT).
- Control the use of the IBM Sysplex Timer (9037), which ensures accurate sequencing and serialization of events.

The values for CLOCKxx are described in *z/OS MVS Initialization and Tuning Reference*. When assigning values for a sysplex configuration, keep in mind:

- Code ETRMODE YES for each system in the multisystem sysplex that is to use the 9037 Sysplex Timer. **Configurations that support ETRMODE NO are XCF-Local mode and Monoplex mode.**
- ETRMODE YES causes the system to ignore the OPERATOR parameter.
- If all members of your sysplex will run in LPARs or under VM on the same side of a single physical processor, and you are using a simulated Sysplex Timer, specify SIMETRID (nn) instead of or along with ETRMODE YES. In no case, specify SIMETRID (nn) with ETRMODE NO. The system will assume that the installation does not have any type of Sysplex Timer.

SIMETRID and OPERATOR PROMPT are mutually exclusive. Specifying both causes the system to reject the CLOCKxx member.

For more information about STPMODE settings, see *Server Time Protocol Planning Guide* (www.redbooks.ibm.com/redbooks/pdfs/sg247280.pdf).

Planning the XCFPOLxx member in parmli

XCFPOLxx can be used only in a multisystem sysplex on a processor with the PR/SM feature.

Functions provided through XCFPOLxx can provide high availability for multisystem applications on an MVS system on PR/SM.

The sysplex failure management (SFM) policy includes all the function available through XCFPOLxx. If a system is connected to an SFM couple data set in which the SFM policy is started, all XCFPOLxx specifications on that system are ignored, regardless of whether the SFM policy is active in the sysplex.

For more information about XCFPOLxx, see *Chapter 8, “Planning sysplex availability and recovery,”* on page 177.

Planning the IEASYMxx member in parmli

IEASYMxx does the following for each system:

- Defines static system symbols
- Specifies the IEASYSxx parmli members that the system is to use.
- Specifies the system name, which is also the substitution text for the &SYSNAME system symbol.

In IEASYMxx, you identify individual systems and, for each system, you can define a unique substitution text for the same system symbols. When the system symbols are specified in shared resource names, each system can substitute unique text for those system symbols.

Specifying the SPINRCVY statement in the EXSPAT parmli member

The EXSPATxx member of parmli allows you to specify actions to be taken to recover, without operator involvement, from excessive spin conditions. The SPINRCVY statement allows you to specify the actions to be taken to end a spin loop.

When adjusting the SPINRCVY specifications in a sysplex, it is important to understand its relationship to the failure detection interval (INTERVAL in the COUPLExx parmli member). (For more information on INTERVAL, see “Planning the failure detection interval and operator notification interval” on page 178.) The default (and recommended) value for INTERVAL, or spin failure detection interval, is computed as follows:

$$(N+1) * SPINTIME + 5 \text{ (seconds)}$$

where N is the number of SPINRCVY actions, and +1 indicates the implicit SPIN action

In an excessive spin condition, the system performs a spin followed by the actions specified on the SPINRCVY statement. The default SPINRCVY specifications are ABEND, TERM, ACR, and the implicit SPIN. Using the default values for INTERVAL and SPINRCVY, the system has time to perform all recovery actions before the INTERVAL time expires.

Changes to EXSPATxx parameters cause new spin failure detection interval to be computed, and then cause the effective failure detection interval to change.

Preparing to use HyperPAV storage subsystems using the HYPERPAV statement in the IECIOSxx parmli member

The HYPERPAV=YES|NO statement in parmli member IECIOSxx lets you dynamically integrate HyperPAV-capable storage servers into your production sysplex. The recommended way to do this is as follows:

1. Apply all HyperPAV maintenance, but wait until step “4” on page 24 to IPL the maintenance.
2. Make sure that all systems in sysplex are in operation with HYPERPAV=NO in effect in parmli member IECIOSxx. Note that the default is HYPERPAV=NO, so that if you do not specify HYPERPAV, you will get HYPERPAV=NO.
3. Install new HyperPAV-capable storage servers or upgrade existing storage servers with HyperPAV capability. Following an upgrade, the HyperPAV License Feature must be activated using either the DS8000® command line interface or the graphical user interface, which is the suggested way.
4. Activate HyperPAV on a single system in the sysplex. Then IPL the HyperPAV maintenance, with an IECIOSxx parmli member that includes a HYPERPAV=YES keyword. You can also dynamically activate HyperPAV using the SETIOS HYPERPAV=YES system command.

Note that if you use the SETIOS HYPERPAV=YES command, it should be performed during periods of low system utilization, as it may take considerable time to convert all capable control units to HyperPAV mode.

If you used SETIOS to go to HYPERPAV=YES mode, you should update the IECIOSxx parmli member to HYPERPAV=YES if you want to remain in HyperPAV mode across future IPL's.

5. Perform step “4” on page 24 to activate HyperPAV mode on the other systems in the sysplex.
6. When all systems in the sysplex are operating in HyperPAV mode, you can analyze the converted control units. You can use RMF or a similar product to determine the proper number of alias devices for each control unit. During peak intervals, RMF shows peak alias usage information for each HyperPAV subsystem. With this information, you can begin to reduce the number of aliases configured for each control unit. Aliases can be added or deleted using the Hardware Configuration Dialog. Note that all systems must be HyperPAV-capable before you reduce the number of aliases.

HyperPAV operations

Use the following commands for HyperPav operations in your sysplex:

- Use the D M=DEV command to determine the number of aliases that are configured in a control unit's alias pool.
- Use the D M=DEV(*devnum*), where *devnum* is a base HyperPAV volume to display information about device *devnum* and the total number of aliases configured in the pool.

- Use `D M=DEV(devnum)`, where *devnum* is a HyperPAV alias device to display information on the HyperPAV alias.

For example, [Figure 2 on page 25](#) shows the information that can be returned by the `D M=DEV` command.

```
SY1 D M=DEV(710)
SY1 IEE174I 14.43.54 DISPLAY M 762
DEVICE 0710 STATUS=ONLINE
CHP          10    20    30    40
DEST LINK ADDRESS 10    20    30    40
PATH ONLINE      Y    Y    Y    Y
CHP PHYSICALLY ONLINE Y    Y    Y    Y
PATH OPERATIONAL  Y    Y    Y    Y
MANAGED          N    N    N    N
CU NUMBER        0700 0700 0700 0700
MAXIMUM MANAGED CHPID(S) ALLOWED: 0
DESTINATION CU LOGICAL ADDRESS = 07
SCP CU ND        = 002107.000.IBM.TC.119B9E00FF07.00FF
SCP TOKEN NED     = 002107.900.IBM.TC.119B9E00FF07.0700
SCP DEVICE NED    = 002107.900.IBM.TC.119B9E00FF07.0710
HYPERPAV ALIASES IN POOL 4
```

Figure 2. Example: Results from D M=DEV command

You can determine the current HYPERPAV setting on the system using the `D IOS, HYPERPAV` command. [Figure 3 on page 25](#) shows an example of the command results.

```
SY1 d ios,HYPERPAV
SY1 IOS098I 14.40.52 HYPERPAV DATA 751
HYPERPAV MODE IS SET TO YES
```

Figure 3. Example: Results from D IOS,HYPERPAV command

Note that the presence of non-FICON channels to a HyperPAV-capable control unit will cause the system to operate on base devices in that logical subsystem (LSS) in base only mode. In base only mode, it is not possible for the system to use alias devices for bases in the logical control unit. You might receive message IOS166E indicating that the HYPERPAV LSS for a device is now in base only mode. You can also detect the condition using the `D M=DEV(bbbb)` command where *bbbb* is a base device in the logical control unit. Once this condition is resolved by removing or configuring offline the non-FICON channels from the HyperPAV configuration, HyperPAV aliases might be usable. However, might have to force the system to rediscover aliases that were not previously discovered using the `VARY` command, where *bbbb* is an online base device in the logical control unit:

```
VARY bbbb,ONLINE,UNCOND
```


Chapter 3. Planning the couple data sets

A sysplex requires a sysplex couple data set to store information about its systems, the XCF groups and members running in the sysplex, and general status information. And, depending on the policies you define to help manage resources and workload for the sysplex, you might need to define additional couple data sets to store policy or function-related information.

- The coupling facility resource management (CFRM) couple data set holds the CFRM policy, which allows you to define how MVS is to manage your coupling facility resources.
- The sysplex failure management (SFM) couple data set holds the SFM policy, which allows you to define how system failures, signaling connectivity failures, and PR/SM reconfiguration actions are to be managed.
- The workload management (WLM) couple data set holds the WLM policy, which allows you to define service goals for workloads.
- The automatic restart management (ARM) couple data set holds the policy that defines how MVS is to manage restarts for specific batch jobs and started tasks that are registered as elements of automatic restart management.
- The system logger (LOGR, LOGRY and LOGRZ) couple data set holds the policies that allows you to define log stream or structure definitions.
- The OMVS (BPXMCDs) couple data set contains system usage and mount information.

Each of these types of functional data, along with real-time information about the sysplex and the resources being managed when the function is in use, resides in a couple data set. Policy specifications and function-specific data cannot reside in the sysplex couple data set, but you can combine data for different functions in the same couple data set.

Before you can define and activate a policy or use this function-specific data, you must format a couple data set to hold the policy, data, or both; you must also ensure that the data set is available to the systems in the sysplex that need it. This topic describes what you need to consider when planning couple data sets for a sysplex.

Many of the things you need to consider when planning couple data sets are common to all couple data sets. However, requirements for connectivity, formatting, and definition differ between the sysplex couple data set and the couple data sets that hold function-specific information. Information about these topics is presented, as follows:

- [“Considerations for all couple data sets” on page 27](#)
- [“Considerations for a sysplex couple data set” on page 32](#)
- [“Considerations for function couple data sets” on page 37](#)

Considerations for all couple data sets

The following considerations apply to sysplex couple data sets as well as to couple data sets that hold policy data.

- **A Couple Data Set Can Be Formatted with the Couple Data Set Format Utility**

IBM provides a couple data set format utility, IXCL1DSU, to format the couple data sets. Sample JCL to invoke the format utility, along with explanations of the information you need to specify when formatting each type of couple data set is provided in [Chapter 11, “Format utility for couple data sets,” on page 313](#).

(The WLM couple data set can also be formatted interactively using the WLM administrative application. See *z/OS MVS Planning: Workload Management*.)

- **Format the couple data set to support the number of systems in the sysplex**

All systems in a sysplex that use function-specific data must be able to access both the primary and alternate couple data sets that were created to contain that data. The value specified on the MAXSYSTEM parameter for function couple data sets should match the value specified for MAXSYSTEM when formatting the sysplex couple data set. This is true for system logger single-system scope LOGRY and LOGRZ couple data set types.

- **An Alternate Couple Data Set Is Highly Recommended**

To avoid a single point of failure in the sysplex, IBM recommends that, for all couple data sets, you create an alternate couple data set on a different device, control unit, and channel from the primary.

Note: For storage subsystem devices like the 2105 ESS, which emulates multiple logical control units, you should attempt to place the alternate couple data set on a different physical control unit if possible.

Information about the sysplex and the services that use couple data sets is maintained in both the primary and alternate couple data sets concurrently. If the primary data set fails, the sysplex automatically makes the alternate the primary.

Besides preventing a single point of failure, having an alternate couple data set has other advantages. You can issue the SETXCF COUPLE,PSWITCH command to dynamically make the alternate couple data set the primary. Thus, in a non-disruptive way, you can:

- Expand the size of the primary couple data set by switching to an alternate that is formatted to accommodate additional ITEMS appropriate to that type of couple data set.
- Change the couple data set to a different device, by defining the alternate couple data set on a different device from the primary. Thus, you can circumvent device errors or take advantage of the characteristics of a different device.

Then, take a formatted spare couple data set (or format a new alternate couple data set) and define it to XCF:

```
SETXCF COUPLE,ACOUPL=(alt-dsname[,alt-volume]),TYPE=dstype
```

The system issues message IXC267E when the primary or alternate couple data set of any type is removed, to indicate that the sysplex is operating without an alternate. Since the remaining couple data set constitutes a single point of failure, consider defining automation rules to respond to message IXC267E and automatically activate a pre-formatted spare couple data set using the SETXCF COUPLE,ACOUPL command.

- **An Alternate Couple Data Set Must Be At Least As Large As the Primary**

After a primary couple data set has become active in the sysplex, the couple data set can be removed from service by adding an alternate couple data set (if one does not currently exist) to the sysplex and by issuing a “PSWITCH” to make the alternate couple data set the primary. When an alternate couple data set is added to the sysplex, it must be at least as large as the current primary data set. That is, the alternate couple data set must be formatted with ITEM values equal to or greater than the ITEM values that were used to format the primary couple data set. For example, if the primary sysplex couple data set was formatted for 50 XCF groups and 200 XCF group members, the new sysplex couple data set must be formatted for 50 or more XCF groups **and** 200 or more XCF group members. Likewise, the values specified for the MAXSYSTEM parameter and if applicable, the MAXGROUP/MAXMEMBER parameters, must be equal to or greater than the values that were used to format the primary couple data set.

When formatting couple data sets consider specifying parameter values commensurate with your hardware and software configuration. You might wish to add some contingency (for example, 20%) to individual parameter values to accommodate growth.

- **Avoid Overspecifying Parameter Values**

Do not excessively overspecify parameter values when formatting couple data sets. Formatting couple data sets with parameters that are overspecified results in wasted space and degraded couple data set performance. Furthermore, if the current primary couple data set is formatted with parameter values that are overspecified and it becomes necessary to format and switch to a couple data set with smaller

parameter values, the sysplex might need to be reinitialized in order to switch to the smaller format couple data set.

It is legitimate to add some contingency to parameter values to allow for growth. However, do not add excessive contingency. Remember that you can always increase couple data set capacity by issuing “PSWITCH” to switch to an alternate couple data set that is formatted with parameter values greater than the parameter values that were used to format the primary couple data set.

- **A Spare Couple Data Set Is Recommended**

When an alternate couple data set replaces a primary, the original primary data set is deallocated, and there is no longer an alternate couple data set.

Because it is advisable to have an alternate couple data set always available to be switched, for each couple data set you plan to use, consider formatting three data sets before the IPL. For example, for the sysplex couple data set, you might format the following:

```
SYS1.XCF.CDS01 - specified as the primary couple data set
SYS1.XCF.CDS02 - specified as the alternate couple data set
SYS1.XCF.CDS03 - a spare
```

Then, if the alternate (CDS02) becomes the primary, you can issue the SETXCF COUPLE,ACOUPLE command to make the spare data set (CDS03) the alternate.

- **The Couple Data Set Must Not Exist Prior to Formatting**

The format utility cannot use an existing data set. This prevents the accidental re-formatting of an active couple data set. You must delete an existing couple data set before reformatting it.

- **The Couple Data Set Must Be Formatted, Not Copied**

The contents of an existing, in-use couple data set should not be copied to another data set for use by another sysplex, or another instance of a sysplex. If a couple data set must be moved to another volume, the preferred way to do this would be to use the IXCL1DSU couple data set format utility to format a new, empty couple data set of the required type on the required volume (making sure that the formatting parameters for the data set provide enough capacity to bring the couple data set into use with the currently-active primary couple data set of that type), and then bring the data set into use as the alternate couple data set using the SETXCF COUPLE,ACOUPLE command. Then, if you want, the new data set might be made the active primary couple data set using the SETXCF COUPLE,PSWITCH command.

Note that couple data sets contain the following:

- The sysplex name of the sysplex to which they pertain and in which they will be used.
- Sysplex ownership records that contain information about the instance of the sysplex that last used the couple data set.
- Various real-time status and policy information that pertains to the function for which the couple data set is formatted, for example, SYSPLEX, CFRM, SFM, and so forth.

Simply making a copy of a couple data set creates a snapshot of all of this information at a point in time; as time then passes, this snapshot of the function's usage and policy information can become more and more out-of-date. Subsequently bringing this couple data set copy into use as a primary couple data set in a sysplex (for example, by initializing a sysplex specifying the couple data set copy as a PCOUPLE) can cause serious problems, as the status information in the couple data set copy is an obsolete snapshot rather than the most up-to-date sysplex information, and might experience inconsistencies with more current information stored in databases or other types of data sets that the sysplex is using.

When re-initializing a sysplex, the most current couple data sets previously in use in the sysplex should always be used. Regressing to use older copies of couple data sets, or couple data sets from another sysplex instance, is not recommended.

For additional information and recommendations regarding the use of remote copy technologies (for disaster recovery) to replicate couple data sets, see [“Planning disaster recovery actions” on page 197](#).

- **A Multiple Extent Couple Data Set Is Not Supported**

For the sysplex couple data set, the format utility determines the size of the data set based on the number of groups, members, and systems specified or defaulted, and allocates space on the volume specified for the data set. There must be enough contiguous space available on the volume for the sysplex couple data set.

For the couple data sets that hold policy information, the format utility determines the size of the data set based on parameter values within the policy.

- **A Couple Data Set Cannot Span Volumes**

XCF does not support multiple-volume data sets.

- **A Couple Data Set Can Be Used by Only One Sysplex**

The name of the sysplex for which a data set is intended must be specified when the data set is formatted. The data set can be used only by systems running in the sysplex whose name matches that in the couple data set.

- **A Couple Data Set Can Share a Volume with Other Data Sets**

However, if you decide to format a couple data set on a volume with other data sets,

- Avoid a volume that has the RESERVE macro issued against it.
- Avoid a volume with high use data sets, such as page or spool data sets.

- **Couple data sets do not support EAV (extended address volumes)** Extended address volumes are not supported for couple data sets. Couple data sets are relatively small and EAV storage is not needed.

- **Performance and Availability Considerations**

The placement of couple data sets can improve performance, as well as availability. For maximum performance and availability, each couple data set should be on its own volume. While this is an expensive approach, placing some data sets on different volumes from others makes sense.

- The primary sysplex couple data set should be on a different volume from the primary CFRM couple data set.
- All other primary couple data sets can reside on one volume, and all other alternate couple data sets can reside on another volume, as shown in the following table. However, be sure to monitor these data sets, and consider placing any high-activity data set on its own volume. (For example, the LOGR couple data set might be a candidate for its own volume if it experiences sufficient I/O activity to warrant its placement on a volume separate from the sysplex or CFRM primary couple data set.)

Volume X Couple Data Sets	Volume Y Couple Data Sets
Sysplex (Primary)	Sysplex (Alternate)
CFRM (Alternate)	CFRM (Primary)
SFM (Primary)	SFM (Alternate)
WLM (Primary)	WLM (Alternate)
ARM (Primary)	ARM (Alternate)
LOGR (Primary)	LOGR (Alternate)
...and so forth	...and so forth

- Place couple data sets on volumes that are attached to cached control units with the DASD fast write (DFW) feature. Although this recommendation applies to all couple data sets in any size sysplex, it is more critical the more systems you have in the sysplex. Those couple data sets most affected are the sysplex couple data set and the CFRM couple data set.
- Place couple data sets on volumes that are not subject to reserve/release contention or significant I/O contention from sources not related to couple data sets. This is true even if the I/O contention is sporadic.
- Do not excessively overspecify parameter values when formatting couple data sets.
- Some sysplex monitoring tools may generate a large amount of I/O to the CFRM couple data set, which may affect system performance. When using this type of tool, be aware of the performance implications. If you are designing a sysplex monitoring tool, see [z/OS MVS Programming: Sysplex Services Reference](#) and [z/OS MVS Programming: Sysplex Services Guide](#) for additional information about how XCF and XES macro services interact with the CFRM active policy.

If a system cannot access a couple data set for an extended period of time (for example, if the volume on which it resides is reserved by another system), MVS switches to its alternate couple data set. To minimize sysplex disruption, when you use DFDSS (or another data mover) to back up a volume with a couple data set, it is recommended that you:

- Convert the SYSVTOC reserves on the volume to global enqueues by creating an entry in the RESERVE conversion RNL for QNAME(SYSVTOC) RNAME(volser). For more information, see [z/OS MVS Planning: Global Resource Serialization](#).
- Use logical volume backup processing so that the backups are done on a data set level, rather than on a track-image level. For more information, see [z/OS DFSMSdss Storage Administration](#).

The system drives periodic I/O to all couple data sets, which acts to verify their continued accessibility. In the case of the sysplex couple data sets, the normal system status updates serve this purpose. For other couple data sets, which are not otherwise frequently accessed, the system "probes" about once per minute to determine whether the data set can still be accessed. Any data set found to be inaccessible due to I/O or device error is removed from service in accordance with XCF's normal recovery protocols. This ensures that problems are surfaced in a timely manner, preventing an undetected single point of failure and allowing the installation to provide an accessible replacement.

• Security Considerations

It is the responsibility of the installation to provide the security environment for the couple data sets. Consider protecting the couple data sets with the same level of security as the XCF address space (XCFAS).

[z/OS Security Server RACF System Programmer's Guide](#) explains how to add a started procedure to the started procedures table.

• The ADRDSSU Utility Can Dump the Contents of a Couple Data Set

If you experience problems with a couple data set or with the usage of a couple data set, you can use ADRDSSU to dump the data set. For example, if you are having problems with coupling facility structures, use ADRDSSU to dump the CFRM data set.

The following JCL could be used to dump a couple data set named SYS1.PRIMARY, which resides on the 3380 volume, SHR001.

```
//DUMP JOB MSGLEVEL=(1,1)
//STEP1 EXEC PGM=ADRDSSU,REGION=4M
//SYSPRINT DD SYSOUT=*
//DD1 DD DISP=SHR,VOL=SER=SHR001,UNIT=3380
//SYSIN DD *
        PRINT DATASET(SYS1.PRIMARY) INDDNAME(DD1)
/*
```

• A Couple Data Set Can Be Defined on an SMS-Managed Volume

You can define a couple data set on a DASD volume that the Storage Management Subsystem (SMS) manages. When placing a couple data set on an SMS-managed volume, consider the following:

- Ensure that all systems that need to can access the couple data set. Catalog both the primary and alternate couple data sets. If the catalog is not shared by all systems, ensure that the couple data set is cataloged on each system. IBM recommends that you catalog the couple data set in the master catalog, rather than a user catalog. If you catalog the couple data set in a user catalog, the volsers must be specified in COUPLExx in order for the data sets to be found during IPL, so there is no benefit in using a user catalog.
- To prevent a single point of failure, define the primary and alternate couple data sets to different control units and different volumes.

The couple data sets can reside on SMS-managed and non-SMS-managed volumes. For example, the primary couple data set can reside on an SMS-managed volume while the alternate can reside on a non-SMS-managed volume.

- Using the Interactive Storage Management Facility (ISMF) panels, define a storage class for the SMS-managed volume with GUARANTEED SPACE=YES. When you use the format utility to format the primary and alternate couple data sets, you can specify the appropriate storage classes (STORCLAS).

You must ensure that the volumes you choose for the couple data set are available to all systems that require it. Specifying the guaranteed space attribute for the storage class ensures that SMS can select from these eligible volumes.

Protect the couple data set from being migrated or deleted by DFHSM. To prevent accidental expiration or migration of the volume by DFHSM, define the following management class attributes through the ISMF panels:

- EXPIRE AFTER NON-USAGE=NOLIMIT
- EXPIRE AFTER DATE/DAYS=NOLIMIT
- COMMAND or AUTO MIGRATE=NONE

You can use the format utility to format the couple data set on an SMS-managed volume by specifying the STORCLAS, MGMTCLAS, or VOLSER keywords. For a description of these keywords, see [Chapter 11, “Format utility for couple data sets,”](#) on page 313.

For information about SMS storage classes and management classes, see *Storage Management Subsystem Migration Planning Guide*.

• Use of Remote Copy Technology with Couple Data Sets

See [“Planning disaster recovery actions”](#) on page 197 for considerations on the use of remote copy technology with couple data sets.

Considerations for a sysplex couple data set

The following considerations apply only to the sysplex couple data set.

Formatting a sysplex couple data set

If you are migrating from an MVS/ESA SP Version 4 system, and you intend to use one or more policies or install more than eight systems in the sysplex, then you must reformat the sysplex couple data set using the OS/390® format utility.

Policy specifications cannot reside in the sysplex couple data set.

So that MVS can determine the size requirements for the sysplex couple data set, you need to specify, in the couple data set format utility, the maximum number of:

- Systems in the sysplex
- XCF groups in the sysplex
- The number of members in the XCF group that has the most members.

To specify the size requirements for your sysplex couple data set, both for your current needs and to provide for future growth, add up all the XCF groups that multisystem applications need, as well as all the XCF groups MVS components need. Then, to the total number of XCF groups, add a contingent number for future growth. The minimum number of groups that you can specify is 10, the maximum is 2045.

Then, determine which group has the largest number of members and specify this number. This maximum number includes all of the group's members that can exist on all systems in the sysplex. The minimum number of members that you can specify is 8, the maximum is 2047. [“Group Names and members for MVS components and subsystems”](#) on page 32 includes a list of MVS components, their sysplex group names, and how to determine the number of members in each group.

Message IXC700E alerts you when the sysplex couple data set is filling up.

Group Names and members for MVS components and subsystems

The following topics contain some examples of XCF groups that might exist in a sysplex.

APPC

APPC has a unique group for each system in the sysplex, and only one system is in each group. The system generates the group name, which is in the format SYSATBxx. The APPC Component Control Block, ATBAPPCA, contains the system's group name.

Console services

Console services has two groups named as follows:

- SYSMCS - has five members plus one member per system. For example, in a sysplex of four systems, console services group SYSMCS has nine members.
- SYSMCS2 - has the number of members calculated as follows:

```
SYSMCS2 members = 1 + n + m

Where:      n = number of systems in the sysplex
            m = RMAX/256 + 1 (round m to the nearest whole number.)

RMAX is the value specified on the RMAX keyword of the
DEFAULT statement in the CONSOLxx parmlib member.
```

For example, in a sysplex of four systems and an RMAX value of 500, group SYSMCS2 has eight members. (SYSMCS2 = 1 + 4 + 3 = 8)

DAE

DAE has one group. The group has one member plus one member for each system sharing the DAE data set in the sysplex. For example, a three system sysplex would have four members in the SYSDAE group. The group name is SYSDAE.

DFSMS and PDSE sharing

For PDSE sharing, DFSMS has two groups named as follows:

- SYSIGW00
- SYSIGW01

Each group has the number of members calculated by the following:

```
members = 1 + n
Where:  n = number of systems in the sysplex with DFSMS installed
```

For example, in a sysplex of four systems that have two systems installed with DFSMS, SYSIGW00 and SYSIGW01 each have three members (SYSIGW00 = 1 + 2 = 3 and SYSIGW01 = 1 + 2 = 3).

ENF

ENF has one group. The group has one member for each system in the sysplex that is running OS/390 release 2 or higher. The group name is SYSENF.

Global resource serialization

Global resource serialization has the following groups when in RING and STAR mode:

- The SYSGRS group with one active member for each system in the sysplex. The members in this group are used for GRS inter system communications and the status of GRS within the system.
- The SYSGRS2 group with one member for each sysplex. The members of the SYSGRS2 group are used to manage the ISGLOCK structure including the migration from RING to STAR mode.

IOS

IOS has one group for each LPAR Cluster in the sysplex. (An LPAR Cluster consists of all systems on a processor that belong to the same sysplex.) The group has one member per system in the LPAR Cluster that is running z/OS V1R1 or higher in zArchitecture mode. The system generates the group name, which

is in the format SYSIOSxx. The "D IOS,GROUP" command can be used to display the IOS group names and the systems that belong to each group.

JES2 multi access spool (MAS)

A JES2 MAS must be entirely contained within one sysplex; including more than one MAS in a sysplex is not recommended.

A MAS in a sysplex must have a unique XCF group name. The default JES2 XCF group name is the local node name defined on the NAME parameter of the local NODE(nnnn) initialization statement. IBM recommends that the default name be used, unless it conflicts with an existing XCF group name. The JES2 XCF group can contain up to 32 members.

Note: If there is more than one JES XCF group in a sysplex, automatic restarts may be affected. See [“Controlling job availability and recovery through automatic restart management” on page 193](#) for more information.

JES3 complex

A JES3 complex must be entirely contained within a sysplex; including more than one JES3 complex in a sysplex is not recommended. A JES3 complex in a sysplex must have a unique XCF group name. The default JES3 XCF group name is the node name defined by the NAME= keyword on the NJERMT initialization statement for the home node (HOME=YES) specified, if one exists. If you have not defined any NJERMT statements, the default is N1. It is recommended that you use the default name, unless it conflicts with an existing XCF group name. You can use the XCFGRPNM keyword of the OPTIONS initialization statement to override the default JES3 XCF group name.

The JES3 XCF group contains one member for each JES3 system active in the JES3 complex, plus one member for each converter/interpreter (C/I) or writer functional subsystem (FSS) active in the JES3 complex.

RRS

RRS has one group. The group has one member for each system in the sysplex. The group name is ATRRRS.

VLF

VLF has one group. The group has one member for each system in the sysplex. The group name is COFVLFNO.

WLM

WLM has one group. The group has one member per system in the sysplex that is running OS/390 or MVS/ESA SP Version 5. The group name is SYSWLM.

XCF

XCF has one group. The group has one member for each system in the sysplex. The group name is SYSXCF. The group is primarily used for sending signals that enable detailed member information to be shown for any active group member.

XES

For structure connections, XES requests one group for each serialized list or lock structure that it connects to. The name of each group is IXCLOxxx, where xxx is a printable hex number. There is one member per structure connection.

Defining a sysplex couple data set to MVS

You define sysplex couple data sets to the system on the PCOUPLE and ACOUPLE keywords of the COUPLE statement of the COUPLExx parmlib member, for example,

```
COUPLE PCOUPLE(SYS1.XCF.CDS01)
      ACOUPLE(SYS1.XCF.CDS02)
```

IBM recommends that you define at least two COUPLExx members that specify different sysplex couple data sets. Then, if the primary sysplex couple data set fails before or during IPL, another COUPLExx member is available for the operator to specify.

Connectivity requirements for a sysplex couple data set

A sysplex couple data set must have full connectivity to all systems in the sysplex.

Sysplex couple data sets and joining a sysplex

A system can IPL into the sysplex if either of the following is true:

- Its COUPLExx parmlib member specifies, on PCOUPLE and ACOUPLE keywords, sysplex couple data sets that match those of the active sysplex.
- Its COUPLExx parmlib member specifies a sysplex couple data set that matches one previously used by at least one of the active systems in the sysplex.

MVS issues messages to a system that IPLs with COUPLExx specifications for couple data sets that do not match those currently in effect in the sysplex. To eliminate these messages, edit the COUPLExx member to reflect the current primary and alternate couple data sets in the sysplex. If sysplex primary or alternate data sets need to be deleted, first delete their specifications from each system's COUPLExx parmlib member.

Sample scenario

The following scenario illustrates how XCF processes COUPLExx data set definitions for systems that are joining the sysplex. Consider that the sysplex has defined four sysplex couple data sets: SYS1.CDS01, SYS1.CDS02, SYS1.CDS03, and SYS1.CDS04.

1. System SYSA initializes the sysplex with the following COUPLExx values:

- PCOUPLE=SYS1.CDS01
- ACOUPLE=SYS1.CDS02

The active primary sysplex couple data set for the sysplex is SYS1.CDS01. The alternate is SYS1.CDS02.

2. System SYSB tries to join the sysplex with the following COUPLExx values:

- PCOUPLE=SYS1.CDS01
- ACOUPLE=SYS1.CDS02

The COUPLExx definitions match those that are in effect for the sysplex. XCF allows SYSB to join the sysplex.

3. The operator issues SETXCF COUPLE,PSWITCH to make the alternate sysplex couple data set SYS1.CDS02 the new primary sysplex couple data set.

The active primary sysplex couple data set for the sysplex is now SYS1.CDS02. There is no alternate defined for the sysplex.

4. The operator issues the SETXCF COUPLE,ACOUPLE command to define SYS1.CDS03 as the new alternate sysplex couple data set for the sysplex.

The active primary sysplex couple data set for the sysplex is SYS1.CDS02. The alternate is SYS1.CDS03.

5. System SYSC tries to join the sysplex with the following COUPLExx values

- PCOUPLE=SYS1.CDS01
- ACOUPLE=SYS1.CDS02

The COUPLExx definitions do not match those that are in effect for the sysplex. XCF issues messages to indicate the values for the sysplex couple data sets in effect for the sysplex. XCF allows SYSC to join because at least one of its sysplex couple data sets was previously defined for the sysplex when SYSA initialized the sysplex.

6. The operator issues SETXCF COUPLE,PSWITCH to make the alternate XCF data set SYS1.CDS03 the new primary sysplex couple data set.

The active primary sysplex couple data set for the sysplex is now SYS1.CDS03. There is no alternate defined.

7. System SYSD tries to join the sysplex with the following COUPLExx values

- PCOUPLE=SYS1.CDS01
- ACOUPLE=SYS1.CDS02

XCF definitions do not match those that are in effect for the sysplex. XCF issues messages to indicate the values for the sysplex couple data sets in the sysplex. XCF allows SYSD to join because at least one of its XCF couple data sets was previously defined for the sysplex when SYSA initialized the sysplex.

8. The operator issues SETXCF COUPLE,ACOUPLE to define SYS1.CDS04 as the new alternate sysplex couple data set.

The active primary sysplex couple data set for the sysplex is now SYS1.CDS03. The alternate is now SYS1.CDS04.

9. The system programmer updates COUPLExx for SYSA, SYSB, SYSC, and SYSD to reflect the current PCOUPLE and ACOUPLE values.

10. The system programmer deletes sysplex couple data sets SYS1.CDS01 and SYS1.CDS02.

Implications of the MAXSYSTEM value for the sysplex couple data set

The sysplex couple data set retains information about systems that are currently active in the sysplex and information about systems that were previously active but are now inactive. Beginning with z/OS V1R9, inactive system information is retained even across a sysplex-wide IPL, for the benefit of systems management software that requires a historical view spanning multiple instances of the sysplex.

When a system joins the sysplex and becomes active, it is assigned a "slot" in the sysplex couple data set. The maximum slot number is the MAXSYSTEM value with which the primary sysplex couple data set was formatted. The slot number chosen, in descending order of preference, is:

- 1, if the IPLing system is the only active system
- The slot last used by a system of the same name, if any
- The lowest-numbered empty slot, if any
- The least-recently-used slot representing an inactive system. (In this case, historical systems management information about the inactive system is lost.)

If there are no matching, empty, or reclaimable slots, the incoming system cannot join the sysplex. In this case, the MAXSYSTEM value is too small.

This slot selection algorithm can cause an incoming system to be assigned a higher slot number than expected. For instance, with two systems already active in the sysplex, the next system to IPL might receive slot number 5 rather than 3. Changing the name of a system is one way this can occur, since the sysplex couple data set will retain a slot for the old name. If the MAXSYSTEM value has been chosen to allow for eventual sysplex growth, the slot number assigned can be greater than the maximum number of systems you expect to be active in the sysplex at its current usage.

If a system is assigned a slot number greater than the MAXSYSTEM value with which one of the function couple data sets was formatted, the incoming system will not be able to use that function. For example, suppose the sysplex couple data set is formatted with MAXSYSTEM=16, but the CFRM couple data set is formatted with MAXSYSTEM=12. If a system IPLing into the sysplex is assigned slot number 13, it cannot use the CFRM function, even if there are 12 or fewer systems actually active in the sysplex. For this reason, it is important that all function couple data sets be formatted with the same MAXSYSTEM value used to format the sysplex couple data set.

Considerations for function couple data sets

The following considerations apply to couple data sets that hold policy information.

Formatting couple data sets

To use a couple data set for CFRM, SFM, WLM, LOGR, LOGRY and LOGRZ or automatic restart management in the sysplex, the couple data set must be formatted with the z/OS couple data set format utility. Under most circumstances, the couple data set must be formatted for the same number of systems as the sysplex couple data set. See [Chapter 11, “Format utility for couple data sets,” on page 313](#).

A couple data set can be formatted to hold one or more policies.

All couple data sets can be formatted using the couple data set format utility, IXCL1DSU, which resides in SYS1.MIGLIB. Sample JCL to invoke this utility and an explanation of the information you need to specify when formatting each type of data set is included in [Chapter 11, “Format utility for couple data sets,” on page 313](#).

If your system supports a system-managed process (such as system-managed rebuild in z/OS Release 8), there are additional requirements and coexistence concerns when formatting your CFRM couple data set. See [“Considerations for supporting CFRM functions” on page 319](#).

The couple data set for the WLM policy can also be formatted interactively, using the WLM administrative application. For more information on this application, see [z/OS MVS Planning: Workload Management](#).

Defining function couple data sets to MVS

For a policy to be in effect on a system, the couple data set that contains the policy must be defined on a DASD that is connected to the system and the couple data set must be defined to MVS.

To define couple data sets for policies (CFRM, SFM, WLM, LOGR, LOGRY, LOGRZ and automatic restart management) to MVS, you can use the DATA statement in the COUPLExx parmlib member or, after IPL, the SETXCF COUPLE command.

For example, to activate SFM primary and alternate couple data sets through parmlib, use the DATA statement, as follows:

```
DATA TYPE(SFM)
  PCOUPLE(SYS1.SFM.CDS01)
  ACOUPLE(SYS1.SFM.CDS02)
```

For example, to activate SFM primary and alternate couple data sets with commands, issue:

```
SETXCF COUPLE,PCOUPLE=SYS1.SFM.CDS01,TYPE=SFM
SETXCF COUPLE,ACOUPLE=SYS1.SFM.CDS02,TYPE=SFM
```

Defining and activating policies

Once the couple data sets for a policy are formatted, you can define and activate the policies that are to reside in the data sets. You can define as many policies as the data set is formatted for. You can combine administrative policies for more than one policy (for example, CFRM and SFM) in the same couple data set, as long as you have formatted the data set to support both types of data.

The administrative data utility, IXCMIAPU, which resides in SYS1.MIGLIB, allows you to define, update, and delete CFRM, SFM, system logger, and automatic restart management policies, and to run reports describing those policies. See [Chapter 12, “Administrative data utility,”](#) on page 335.

The WLM administrative application is available to define the WLM policy. See [z/OS MVS Planning: Workload Management](#).

The following topics provide information on defining and activating policies:

- The SFM policy is described in [“Controlling system availability and recovery through the SFM Policy”](#) on page 180.
- The CFRM policy is described in [“Planning a coupling facility policy”](#) on page 49.
- The WLM policy is described in [z/OS MVS Planning: Workload Management](#).
- The automatic restart management policy is described in [“Controlling job availability and recovery through automatic restart management”](#) on page 193.
- The LOGR policy is described in [“Format the sysplex scope LOGR couple data set and make it available to the sysplex”](#) on page 276.
- The LOGRY and LOGRZ policies, see [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex”](#) on page 279.

Updating policies in a sysplex

In some cases you can update or change policy information in an existing policy, while at other times you must format a new couple data set in which to define the updated policy. The difference is determined by the size of the couple data set as it was originally formatted.

A couple data set is formatted to hold not only the policy data but also status data about the resources defined in that policy. Thus, if you increase the number of resources within a policy (for example, increasing the number of structures that can be defined in a policy), it might be necessary to define a new policy in a newly formatted couple data set. In general,

- If the policy update requires increasing the size of the couple data set in which the policy resides, then you must first format a new couple data set using the IXCL1DSU utility.
- If the update does not require increasing the size of the couple data set, then you can use the IXCMIAPU administrative data utility to create your updates.

See [“Formatting couple data sets”](#) on page 37 for information on formatting a new couple data set.

To update policy information dynamically, you must run the administrative data utility, IXCMIAPU, against the active couple data set. The active couple data set contains both the active policy and a set of administrative policies. Run IXCMIAPU, omitting the DSN and VOLSER information, to update one or more administrative policies. XCF automatically synchronizes the updated policy information to the alternate couple data set. To switch from the active policy to an updated administrative policy, issue the SETXCF command to start the new policy.

See [“Coding the administrative data utility”](#) on page 336 for information about updating policies in a sysplex using IXCMIAPU. Also, see [“Updating a CFRM policy”](#) on page 66 for an example of updating a CFRM policy.

Connectivity requirements for function couple data sets

For a function (such as, CFRM, ARM, or SFM) to be used by the sysplex, the couple data set containing the policy and/or data associated with that function must be accessible to all systems that need to use it. If a system cannot access a function couple data set, then the system does not use that function. Any policy associated with that function does not become active on that system.

When formatting a function couple data set, IBM strongly recommends that you use a MAXSYSTEM value equal to the value used to format the primary sysplex couple data set. Using a smaller value can prevent a system from using the function couple data set, as described in [“Implications of the MAXSYSTEM value for the sysplex couple data set”](#) on page 36. Using a larger value results in an unnecessarily large

couple dataset, wastes system storage, and degrades I/O performance. Beginning with z/OS V1R12, the (IBMXCF,XCF_CDS_MAXSYSTEM) health check reports on any function couple data set formatted with a MAXSYSTEM value less than that used for the sysplex couple data set.

The CFRM couple data set must have connectivity to all systems in the sysplex that use the coupling facility. If a system using a CFRM data set loses connectivity to the data set, the system is placed in a non-restartable wait state.

The requirement to format a function couple data set with the same MAXSYSTEM value used for the sysplex couple data set is particularly critical for the CFRM couple data set. If an IPLing system is assigned a sysplex slot number greater than the CFRM MAXSYSTEM value (see [“Implications of the MAXSYSTEM value for the sysplex couple data set”](#) on page 36), one of the following situations can result:

- If sysplex signaling connectivity is only available through coupling facilities, the system will not be able to join the sysplex.
- If the system is IPLing in GRS star mode, the system will enter a non-restartable OA3 wait state.

Providing coupling facility information to a recovery manager

You can use CFRM to communicate site information to a Recovery Manager, such as Geographically Dispersed Parallel Sysplex (GDPS). Include site information in the CFRM policy using the SITE keyword of the Administrative Data Utility. The recovery manager can then notify CFRM of its status and indicate which site is the recovery site. Note that the recovery manager must have been granted UPDATE access authority for the 'MVSADMIN.XCF.CFRM' resource name in the z/OS Security Server FACILITY class. The recovery site information provided by the resource manager assists CFRM when making CF structure duplexing failover decisions.

Chapter 4. Managing coupling facility resources

Sysplex support for a coupling facility satisfies customers' requirements for:

- Data sharing across the systems in a sysplex
- Resource sharing across the systems in a sysplex
- Maintaining the integrity and consistency of shared data
- Maintaining the availability of a sysplex.

The coupling facility makes data sharing possible by allowing data to be accessed throughout a sysplex with assurance that the data will not be corrupted and that the data will be consistent among all sharing users. The coupling facility provides the medium in which users of coupling facility structures can reconstruct shared data should a failure occur and can be used to assist in the isolation of failing systems.

The coupling facility is a shareable storage medium, but not what we usually consider a shared storage device. Rather, a coupling facility is licensed internal code (LIC) running in a special type of PR/SM logical partition (LPAR) in certain z/Series and S/390® processors. A coupling facility can be shared by the systems in one sysplex only; it cannot be shared simultaneously by multiple sysplexes.

The following topics describe what a coupling facility is and how to include one or more coupling facilities in your sysplex:

- [“Planning for a coupling facility” on page 41](#) outlines the steps your installation must perform to use a coupling facility.
- [“Planning a coupling facility policy” on page 49](#) describes how to set up the policy that will govern your use of a coupling facility.
- [“An installation guide to duplexing rebuild” on page 69](#) describes the installation tasks required to enable system-managed duplexing rebuild.
- [“Encrypting coupling facility structure data” on page 80](#) describes how to encrypt customer data while it is being transferred to and from the coupling facility (CF) and while it resides in a coupling facility structure.
- [“The role of CFRM in a sysplex” on page 84](#) explains the services that coupling facility resource management provides to manage a coupling facility in your installation.
- [“Resource allocation in the coupling facility” on page 87](#) describes how the storage in a coupling facility is allocated, based on your requirements and those of the application using a coupling facility.
- [“Operating in a coupling facility environment” on page 89](#) describes the operator intervention that might be required with a coupling facility in a sysplex and summarizes the operator commands with which you interface with a coupling facility.
- [“Coupling facility structures for IBM products” on page 96](#) provides a chart of coupling facility structures that are used by IBM products and where to find additional information about the structures.

Planning for a coupling facility

Using a coupling facility in your sysplex requires both hardware and software. First, you must have a processor that supports the coupling facility control code of the coupling facility. Second, you must have a processor on which one or more MVS images will run which is capable of attaching to the coupling facility with coupling facility links. Third, you must have the appropriate level of z/OS that allows you to manage the coupling facility resources.

The information provided here is an overview of the points to be considered when adding a coupling facility to your sysplex.

Defining a coupling facility

A coupling facility runs as a special type of logical partition (LPAR) on certain IBM Z and zEnterprise servers. See *PR/SM Planning Guide* for a list of all processors that support a coupling facility.

The coupling facility is defined through PR/SM panels. Once you have defined an LPAR to be a coupling facility logical partition, only the coupling facility control code can run in that partition. When you activate the coupling facility LPAR, the system automatically loads the coupling facility control code from the processor controller or the support element of the processor.

In the same manner, once you deactivate the coupling facility LPAR (for example, by doing a power-on-reset of the processor), the system does not preserve all coupling facility control code and any associated user data that might reside in the coupling facility.

Configuring a processor and a coupling facility

Coupling facility channels provide the connectivity between a coupling facility and the CPCs or LPARs directly attached to it. Coupling facility links (ISC-3 and IFB links) are defined to the server running the z/OS operating system and to the LPAR running the coupling facility.

PR/SM Planning Guide describes the different type of coupling facility channels, including peer mode links, which can provide both sender and receiver capability on the same link by being configured as an unshared channel path to a single coupling facility and at the same time as a shared channel path among several sender LPARs. CF-to-CF links allow a coupling facility sender channel to communicate with another coupling facility, a function that is required for system-managed duplexing rebuild.

PR/SM Planning Guide also provides guidance for configuring coupling facility links in a sysplex. Just as a coupling facility cannot be shared among sysplexes, there are configuration restrictions for coupling facility links as well. Links from a coupling facility owned by a sysplex should not be connected to systems outside the sysplex. If a configuration does contain CF links that are connected to systems both in and outside the owning sysplex, the CF links to systems outside the sysplex must be configured offline before the coupling facility is used by systems in that sysplex.

Both the coupling facility LPAR and the coupling facility channel paths must be defined to the I/O configuration data set (IOCDs). HCD provides the interface to accomplish these definitions and also automatically supplies the required channel control unit and I/O device definitions for the coupling facility channels.

To determine the logical partition identifier to specify in the CFRM policy, following these rules:

- If the coupling facility is defined on a z990, z890, or later processor, then the PARTITION value must match the partition ID associated on the activation profile for the CF image on the support element or hardware master console.
- If the coupling facility is defined on a pre-z990 or pre-z890 or earlier processor, then the PARTITION value must match the partition number specified by HCD/IOCP in the IOCDs for that partition and can be in the range of 01-0F.
- When moving a coupling facility to a different PARTITION, remember to update the CFRM policy.

See [“CFRM parameters for administrative data utility” on page 346](#) for information about defining a CFRM policy.

See [z/OS HCD Planning](#) for information about defining a coupling facility with HCD.

See *PR/SM Planning Guide* for information about defining a coupling facility on a z900 processor using the Customize/Delete Activation Profiles task.

Understanding the coupling facility level (CFLEVEL)

The level (CFLEVEL) of the coupling facility control code (CFCC) that is loaded into the coupling facility LPAR determines what functions are available for exploiting applications. Different levels provide new functions and enhancements that an application might require for its operation. As more functions are added to the CFCC, it might be necessary to allocate additional storage to a coupling facility structure.

Similarly, as new functions are added, the coupling facility itself may require additional storage. In any configuration, the amount of fixed storage required for the coupling facility is based on configuration-dependent factors. This “fixed overhead” may be associated with the coupling facility's code areas or fixed controls and must be considered in addition to the storage needed for structure allocation.

Important!

IBM recommends that you review the sizes of all your coupling facility structures using the CF Structure Sizer (CFSizer) tool before migrating to a coupling facility at a higher CFLEVEL.

After reviewing your coupling facility structures with CF Structure Sizer (CFSizer), modify your CFRM policy INITSIZE and SIZE values appropriately for the structures that changed. Then activate the modified CFRM policy before bringing the coupling facility at the new CFLEVEL into active use.

In general, IBM recommends that coupling facility structures be sized generously, erring on the side of too-large rather than too-small, because sizing structures too-small is a major factor in many different kinds of structure-related problems and outages. Be careful to re-plan so that:

- The physical coupling facility LPARs in the configuration have enough storage to accommodate the structures with their new sizes, in aggregate.
- There is enough unused space in the coupling facilities so that if a coupling facility failure occurs, the re-sized structures are still recoverable into other CF(s) in the configuration.

Make physical coupling facility LPAR storage changes if necessary to accommodate this.

See also “[Requesting structure size](#)” on page 50.

PR/SM Planning Guide provides information about the functions available with each CFLEVEL of a coupling facility, model-dependent limits, and other characteristics such as performance, storage allocation, and operations.

Base coupling facility function is available with coupling facility control code (CFCC) level 0 (CFLEVEL=0). Additional function and enhancements are provided by follow-on CFLEVEL versions. For the most accurate list of CFLEVEL functions with associated hardware and software corequisites, see [Coupling Facility Level \(CFLEVEL\) Considerations \(www.ibm.com/docs/en/systems-hardware/zsystems/9175-ME1?topic=considerations-coupling-facility-level-cflevel\)](http://www.ibm.com/docs/en/systems-hardware/zsystems/9175-ME1?topic=considerations-coupling-facility-level-cflevel).

An application might require that its structure be allocated in a coupling facility at a minimum coupling facility level. Or, an application might have a requirement for a particular CFLEVEL, but if a coupling facility of that level is not available, might allow its connectors to connect and run, bypassing the architected function provided by the higher CFLEVEL that was unavailable. To ensure that the structure allocation is successful, you must provide a preference list containing coupling facilities that meet or exceed the minimum level in your CFRM policy.

“[CFLEVEL and operating system level coexistence](#)” on page 43 provides a summary of these functions and some exploiting products.

CFLEVEL and operating system level coexistence

Table 2 on page 44 shows which operating systems can use coupling facilities at different CFLEVELs, either to exploit the function provided by the CFLEVEL or to coexist with the CFLEVEL function without exploiting it.

- An operating system release “coexistence” support row of the table with an APAR number indicates required service in addition to the operating system product code to allow the operating system to use existing operating system and CF functions, but not exploit new functions that are contained in a particular CFLEVEL.
 - For a given z/OS release / CFLEVEL combination, APARs listed in the corresponding cell and all cells to the left of that cell are required for coexistence.
 - A blank area in an operating system coexistence row indicates no additional service is required for the operating system product code to use the CFLEVEL except as noted by the preceding bullet.

- Example: A system at z/OS V2R2 can coexist with a coupling facility up to and including CFLEVEL 21 without additional service, but it requires the service provided by APAR OA52058 to coexist with CFLEVEL 22 and all higher CFLEVELs. Higher CFLEVELs may require additional service as noted in the table. In particular, although there is no service specifically listed for CFLEVEL 23, all APARs listed in corresponding cells to the left of the blank cell are required for coexistence.
- An operating system release “exploitation” support row of the table with text (an ‘X’ or an APAR number) indicates exploitation of the functions contained in a particular CFLEVEL by the operating system release.
 - A blank area in an exploitation row indicates that the z/OS release does not exploit functions provided by that CFLEVEL.
 - Example: A system at z/OS V2R2 can exploit all function provided by CFLEVELs up to and including CFLEVEL 20, requires additional service to exploit CFLEVELs 21-24 functions, and cannot exploit functions provided by CFLEVEL 25.

Fix categories (FIXCATs) are used to identify whether an APAR and its related fixing PTF is required by a z/OS system to coexist with a CFLEVEL associated with a hardware server device, or required to exploit function provided by the CFLEVEL. For example, APAR OA56345 is required by the operating system to run with CFLEVEL 24 in a sysplex configuration, whereas OA56774 is required to exploit a particular CFLEVEL 24 function.

See the IBM Fix Category (FIXCAT) values <https://www.ibm.com/support/pages/ibm-fix-category-values-and-descriptions> that identify fixes that provide z/OS software support for the specific hardware server device associated with the CFLEVEL.

See *PR/SM Planning Guide* for information on hardware server device model support for CFLEVELs.

Table 2. CFLEVEL summary table						
Release	CFLEVEL					
	Support	21	22	23	24	25
z/OS V2R2	Coexistence		OA52058		OA56345	OA60275
	Exploitation	OA47796	OA51862	OA54688	OA56774	
z/OS V2R3	Coexistence				OA56345	OA60275
	Exploitation	X	X	OA54688	OA56774	OA60650
z/OS V2R4	Coexistence					OA60275
	Exploitation	X	X	X	OA56774	OA60650
z/OS V2R5	Coexistence					OA60275
	Exploitation	X	X	X	X	OA60650

Note:

All operating system releases listed in [Table 2 on page 44](#) can exploit all function provided by CFLEVELs up to and including CFLEVEL 20.

The functions that are provided by each CFLEVEL are described briefly. For detailed information, see *PR/SM Planning Guide*.

0

Base coupling facility support

1

Structure alter support

2

New cache and lock structure functions for improved performance by allowing operations on lists of cache entries (batched registration) and lock entries (batched unlock), rather than on individual entries. These functions were intended for initial exploitation by DB2® and IRLM.

- 3** List monitoring enhancements for keyed list structures (“event monitoring”). The enhancements provide monitoring at a finer level of granularity than was available previously. These functions were intended for initial exploitation by IMS shared message queue.
- 4** Alter and dump support for list structures that use event monitoring.
- 5** User-managed duplexing of DB2 group buffer pools for added availability. This function was intended for initial exploitation by DB2.
- 6** Function that is provided for the TPF operating system. There is no new function for OS/390 exploitation, but all previous CFLEVEL function remains available.
- 7** Cache structure name class queue support that allows entries to be deleted more efficiently. This function was intended for initial exploitation by DB2.
- 8** System-managed rebuild support.
- 9** Support for list structure exploitation by WebSphere® MQ.
- 10** CF-to-CF connectivity support (required for CF Duplexing).
- 11** System-managed duplexing support for all coupling facility structure types.
- 12** Performance enhancements for cache structures by allowing batching of write requests, cross-invalidate requests, and castout requests. These functions were intended for initial exploitation by DB2.
- 13** Performance enhancements for cache structures for IXLCACHE REQUEST=READ_COCLASS requests. This function was intended for initial exploitation by DB2.
- 14** Performance enhancements for coupling facility dispatching and latching.
- 15** Performance enhancements for CF duplexing to suppress RTE signals.
Note: This RTE suppression function is not enabled by z/OS.
 Granular CF CP Utilization reporting by structure.
 CF multitasking enhancements (increased number of CF tasks).
- 16** CF duplexing enhancements and shared message queue list notification enhancements.
- 17** Dumping support for CFCC non-disruptive coupling facility. Increased the maximum number of coupling facility structure instances per coupling facility image from 1023 to 2047 and provided support for greater than 32 connectors to a coupling facility list or lock structure. Installations should not deploy more than 32 instances of the application until the following recommendations are met.
 - Upgrade all relevant application instances to a level that supports greater than 32 connectors.
 - Ensure that the sysplex contains at least two coupling facilities that are CFLEVEL=17 or higher.
 Failure to implement the recommendations that were previously stated might result in an unsafe migration path to greater-than-32-connector support for a structure and can lead to failed connection attempts, failure to rebuild the structure, or failure to duplex the structure.

18

Functions include:

- Cache performance and reliability improvements.
- Coupling-related adapter interrupt exploitation.
- Enhanced serviceability information for coupling channels.

19

Storage-class memory.

21

Asynchronous duplexing support.

22

Improvements to the efficiency of scheduling and processing work targeted to system-managed synchronous duplexed CF structures.

To take advantage of this improvement without sacrificing duplexing reliability, you must run z/OS V2R3 or a lower release with a PTF for APAR OA52058 when a CFLEVEL=22 CF is involved in duplexing.

New CF structure list monitoring functions include:

- List full/not-full monitoring
- Aggressive list and key range monitoring notification when entries are added to lists and key ranges
- List and key range monitoring notification delays.

The list monitoring functions are available to list structure connectors running on z/OS V2R3 or on z/OS V2R2 with APAR OA51862 when a structure is allocated in a CFLEVEL=22 or higher CF.

23

Asynchronous Cache Cross-Invalidation (XI) for Coupling Facility Cache Structures

Asynchronous Cache XI is a communication protocol that notifies connectors of the completion of data invalidation asynchronously with respect to the IXLCACHE request.

The Asynchronous Cache XI function is available to cache structure connectors running on z/OS V2R4 or a lower release with a PTF for APAR OA54688 when a cache structure is allocated in a CF at CFLEVEL 23 or higher.

24

- CFCC Fair Latch Manager

Provides improvements to the efficiency of scheduling and processing work targeted to system-managed synchronous duplexed coupling facility structures. z/OS must provide new information to the CF to permit exploitation of system-managed synchronous structure duplexing when either structure instance resides in a CF at CFLEVEL 24.

- CFCC Message Path Resiliency Enhancement

When z/OS recognizes an inconsistency in the system identification information associated with a message path to a CF, z/OS will initiate the capturing of diagnostic information from the CF, channel subsystem, and connected z/OS images, and will then take immediate action to transparently correct the inconsistent information dynamically

- Monopolization Avoidance for CF Tasks

A PTF for APAR OA56774, in conjunction with CFCC CFLEVEL 24 provides new function to prevent a runaway sysplex application from monopolizing a disproportionate share of CF resources

25

With a PTF for APAR OA60650, in conjunction with CFCC CFLEVEL 25, functions include:

- **Read Retry Buffer Extensions**

Read retry buffer support extended to the set of non-idempotent CF cache and lock structure commands, thus reducing the likelihood of encountering indeterminate status results from coupling facility commands.

- **Structure Full Threshold**

Allowance for the designation of a percentage of the total number of lock structure record data entries that must remain free after completion of a command to create a record data entry, in order for a creation operation to proceed. To improve connector application resiliency, such as when recovery actions need to allocate structure resources, lock structure connectors can request that the CF reserve a percentage of resources from general mainline application usage until the application explicitly asks for the reserved resources to be used to satisfy structure requests.

- **Cache Residency Time Metrics**

Availability of cache structure data and directory residency statistics which indicate how long data area and directory entries reside in a cache structure before they are reclaimed. This information can be used to improve structure sizing and apportionment.

The DISPLAY CF command will always display the actual CFLEVEL of the coupling facility. This might differ from what the application understands to be the operational level of the coupling facility. The operational level refers to the architectural level required to perform the necessary operations against the structure.

Specifying coupling facility non-volatility

An application that is using the coupling facility might require non-volatility of the coupling facility storage. Depending on the processor on which the coupling facility is defined, you might have to provide a backup power supply to make the contents of coupling facility storage nonvolatile across utility power failures.

PR/SM Planning Guide describes the processor requirements for coupling facility non-volatility and the coupling facility control code MODE command. The MODE command must be set so that applications using the coupling facility can monitor its non-volatility status.

Planning for coupling facility failure-independence

An application might require that its structure be placed in a failure-independent environment. To accomplish this, you must ensure that the coupling facility is not in the same configuration as the MVS systems that access it. For example, placing the coupling facility in an LPAR in a processor with one or more additional LPARs that are running MVS to access the coupling facility would not provide a failure-independent environment.

Determining your coupling facility requirements

PR/SM Planning Guide provides information for use in determining the level of hardware and CFCC required for your installation:

- CPC model numbers that support a coupling facility
- CFCC EC levels
- Software corequisites
- Correspondence between CFCC levels and CFLEVELs
- Explanation of the functions provided by CFLEVEL

Managing a coupling facility

Storage in a coupling facility is divided up into distinct objects called structures. Structures are used by authorized programs to implement data sharing and high-speed serialization. Structure types are cache, list, and lock, each providing a specific function to the application. Some storage in the coupling

facility can also be allocated as a dedicated dump space for capturing structure information for diagnostic purposes.

You manage a coupling facility through a policy, the coupling facility resource management (CFRM) policy. CFRM allows you to specify how a coupling facility and its resources are to be used in your installation. In a CFRM policy, you supply information about each coupling facility and each coupling facility structure that you plan to use.

Your installation can define several CFRM policies, to be used at different times depending on the workload running on the sysplex. Each of these *administrative* policies has its own name by which you identify it when you want to make it the *active* policy. Only one policy can be active in a sysplex at a time.

You create the administrative CFRM policies with an IBM utility program, IXCMIAPU, provided in SYS1.MIGLIB. Once created, the policies reside in a CFRM couple data set. MVS and the authorized application use the definitions in the CFRM policy when managing the coupling facility storage.

Two ways to start a CFRM policy

There are two ways to start using a CFRM policy in your sysplex:

- You can specify the name of the CFRM policy to be activated at IPL-time in the COUPLExx parmlib member that is used to initialize the sysplex. The system uses the CFRM policy name specified by the CFRMPOL keyword on the COUPLE statement to identify the policy to be started if there is no other previously-activated CFRM policy in effect. Note that the CFRM couple data set containing the policy must be accessible to all members of the sysplex.

The ability to specify a CFRM policy to be started at IPL-time allows you to initialize your sysplex in global resource serialization star mode when there is no previously-activated CFRM policy, if the CFRM policy started contains the ISGLOCK structure required for star mode.

- You can specify the name of the CFRM policy to be started by issuing the

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

command specifying the policy name. Note that this command cannot be issued until the sysplex has completed initialization using the COUPLExx parmlib member.

Authorizing coupling facility requests

As documented by the subsystem or application using a coupling facility structure, the security administrator might have to define security profiles for certain structures. If the z/OS Security Server, which includes RACF®, or another security product is installed, the administrator can define profiles that control the use of the structure in the coupling facility.

The following steps describe how the RACF security administrator can define RACF profiles to control the use of structures:

1. Define resource profile IXLSTR.structure-name in the FACILITY class.
2. Specify the users who have access to the structure using the RACF PERMIT command.
3. Make sure the FACILITY class is active, and generic profile checking is in effect. If in-storage profiles are maintained for the FACILITY class, refresh them.

For example, if an installation wants to permit an application with an identifier of SUBSYS1 to issue the IXLCONN macro for structure-name CACHE1, the security administrator can use the following commands:

```
RDEFINE FACILITY IXLSTR.CACHE1 UACC(NONE)
PERMIT IXLSTR.CACHE1 CLASS(FACILITY) ID(SUBSYS1) ACCESS(ALTER)
SETROPTS CLASSACT(FACILITY)
```

You can specify RACF userids or RACF groupids on the ID keyword of the PERMIT command. If RACF profiles are not defined, the default allows any authorized user or program (supervisor state and PKM allowing key 0-7) to issue coupling facility macros for the structure.

For information about RACF, see [z/OS Security Server RACF Security Administrator's Guide](#).

Planning a coupling facility policy

To set up your CFRM policy, you must know which subsystems and applications in your installation are making use of the coupling facility and what their structure requirements are. Each application specifies the structure types required and provides information as to the name and size of the structure or structures it uses. Based on this input, you must determine what is an appropriate number of coupling facilities and how best to distribute the structures within the coupling facilities.

A single coupling facility can only be used by one sysplex at a time.

To define a CFRM policy, you must:

- Identify each coupling facility.
- Define the amount of space within the coupling facility that you want for dumping.
- Identify each structure. For each structure:
 - Specify the size of each structure, the maximum size and optionally, an initial size and a minimum size.
 - Specify parameters that are related to the use of storage-class memory.
 - Specify a percentage value that represents a percent full threshold for the structure.
 - Specify whether you want the system to automatically alter the size of a structure when it reaches a user-defined or system-defined percent full threshold.
 - Specify which structures you want to duplex.
 - Specify an ordered “preference list” of coupling facilities in which the structure may be allocated.
 - Specify a list of structures, an “exclusion list”, that are not to be allocated in the same coupling facility as the structure.
 - Specify a percent value to indicate the amount of lost connectivity to a structure that the sysplex can tolerate before MVS initiates a structure rebuild, if applicable.

The administrative data utility, IXCMIAPU, and an example of its use, is described in [Chapter 12](#), “Administrative data utility,” on page 335.

Identifying the coupling facility

The coupling facility is identified to CFRM in two ways. First, each coupling facility has a unique 32-byte identifier that consists of a set of discrete variables, such as machine type, manufacturer, plant where manufactured, sequence number, CPC identifier, and configuration code. The values for each of these variables in the unique identifier are displayed on the hardware console when you first define the coupling facility LPAR and the system loads the coupling facility control code. You use these values when defining the coupling facility in a CFRM policy.

Note: This coupling facility identifier in the CFRM policy must be kept consistent with the current configuration. Whenever changes that affect coupling facilities are made to the configuration, the CFRM policy must be inspected to make sure the coupling facility identifiers are correct.

Second, each coupling facility has a logical name — an eight-character name that you assign. It is this shorter, more usable coupling facility name that you use on interfaces with the coupling facility, for operator commands and recovery procedures, for example. The logical name and the unique worldwide identifier are paired in the definition of each coupling facility in a CFRM policy.

PR/SM Planning Guide describes the coupling facility and provides the information required for its installation and operation.

Identifying the coupling facility structures

MVS recognizes three structure types — cache, list, and lock, and provides unique programming services for each of the structure types to allow the manipulation of data within the structure. To use these

services, a system component, subsystem, or application “connects” to the structure, specifying a structure name and the type of structure it is.

z/OS MVS Programming: Sysplex Services Guide and *z/OS MVS Programming: Sysplex Services Reference* describe the programming services provided for coupling facility structures.

In your CFRM policy, you must provide the definitions to allow the application to use the structure. The CFRM policy defines the structures, including the amount of coupling facility storage to be used for each structure. Note that the CFRM policy does not identify the type of structure (cache, list, or lock). That information is provided by the application connecting to the structure. A coupling facility can support multiple coupling facility structures, but any one structure must exist entirely within a single coupling facility.

Requesting structure size

The size of a coupling facility structure includes the total amount of storage required by both the application and by the coupling facility itself for structure control information, and is specified in size units of kilobytes, megabytes, gigabytes or terabytes. Determining the correct size is based on the application's use of the structure, characteristics of the workload environment in which the structure is to be used, and the CFLEVEL of the coupling facility in which the structure is allocated. The application provides guidance so that you can estimate an appropriate size for a structure. The IBM Z Coupling Facility Structure Sizer is also available to help estimate the amount of storage required for a structure. (See [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062) for more details.) You should use RMF reports to monitor the structure's utilization. Also, you can use structure full monitoring to monitor the structure utilization. The combined functions of structure full monitoring and automatic structure altering allow the system to tune the size of the structure when it reaches a full threshold.

The size of a structure can be specified by the application in its code or by the installation in its CFRM policy. The size that you specify in the CFRM policy with the SIZE specification is the requested maximum structure size. (The system might override this maximum size during rebuild processes. See “How MVS initially allocates the structure” on page 50.) You can change the structure size with the ALTER function if the application has specified that ALTER is allowed. ALTER allows you to vary the structure size between an initial size (specified by INITSIZE in the CFRM policy) and the maximum size. The coupling facility determines the minimum structure size, the minimum amount of coupling facility space required to allocate the structure with the specified entry-to-element ratio. It is possible to have the system automatically alter the size of a structure when the system determines that the structure has surpassed a “structure full” threshold. In addition to the application specifying that the structure is able to be altered, the installation must indicate in the CFRM policy that the system is to monitor the structure and automatically alter its size when required. The MINSIZE specification in the CFRM policy indicates the smallest size to which the structure can be automatically altered.

How MVS initially allocates the structure

The system allocates storage for a coupling facility structure based on the level of the system requesting the allocation and the level of the coupling facility in which the structure is to be allocated. The amount of storage allocated is based not only on the connection parameters specified by the application but also on features supported by the coupling facility and the system performing the structure allocation.

Several values can affect the size of a structure — the STRSIZE value specified by the exploiter on the IXLCONN invocation and the SIZE, INITSIZE, and MINSIZE values specified by the installation in the CFRM policy. If a value is too small to satisfy the required control structure space, the connection attempt will fail. As systems migrate to higher level coupling facilities that support additional features, it is possible that a value that was satisfactory for a lower level coupling facility might be unsatisfactory for a higher level.

It is strongly recommended that when you migrate to a higher level (CFLEVEL) coupling facility, you reassess the value specified for each structure's size. Higher level coupling facilities provide new function, which may or may not require additional control storage to be allocated for a structure. Existing structures that might eventually be reallocated or rebuilt in a higher level coupling facility might have additional storage requirements that are not reflected in the size that was determined for a lower level coupling facility.

The MINSIZE specification in the CFRM policy allows an installation to indicate the smallest size to which a structure can be altered, as well as to provide a minimum bound on all structure allocation requests. MINSIZE is an optional parameter in the CFRM policy, and its default value depends on the specification of ALLOWAUTOALT in the CFRM policy.

- If ALLOWAUTOALT(YES) is specified and MINSIZE is not specified, its default value is 75% of the INITSIZE value or 75% of the SIZE value if INITSIZE is not specified.
- If ALLOWAUTOALT(NO) is specified or defaulted to and MINSIZE is not specified, its default value is zero.

Structure size allocation uses the STRSIZE defined on the IXLCONN macro, if the application specified a value, and that value is less than the maximum size specified in the CFRM policy with the SIZE parameter and greater than the MINSIZE value. If STRSIZE is not specified on the IXLCONN macro, allocation uses the INITSIZE specified in the CFRM active policy. If INITSIZE is not specified (it is an optional parameter), then the SIZE specified in the CFRM active policy is used.

Table 3 on page 51 shows the initial allocation size determination for a connection. The STRSIZE value specified on IXLCONN might or might not be present, depending on the application using the structure. The INITSIZE value from the CFRM policy also might or might not be present. The MINSIZE value is assumed to have a non-zero value in the table. In all cases, the initial allocated size of the structure will not be greater than the maximum structure size specified in the CFRM policy with the SIZE parameter nor less than the MINSIZE from the CFRM policy.

<i>Table 3. Initial Structure Size Allocation</i>		
IXLCONN	CFRM Policy	
	INITSIZE specified	INITSIZE not specified
IXLCONN STRSIZE specified between MINSIZE and SIZE	Target size = STRSIZE	Target size = STRSIZE
IXLCONN STRSIZE specified greater than SIZE	Target size = SIZE	Target size = SIZE
IXLCONN STRSIZE specified less than MINSIZE	Target size = MINSIZE	Target size = MINSIZE
IXLCONN STRSIZE not specified	Target size = INITSIZE	Target size = SIZE

To display the actual amount of storage allocated to a structure, issue the DISPLAY XCF,STRUCTURE command.

Structure allocation when rebuilding a structure

The results of structure allocation during the rebuild process, whether user-managed or system-managed, may differ significantly from those during initial allocation, depending on what system conditions exist. The size of a structure undergoing a rebuild or duplexing rebuild process might require more coupling facility storage than specified by the SIZE parameter in order to allocate the new structure. As a general rule, SIZE specifies the maximum size to which a structure can expand. However, because the rebuild process requires that the rebuilt structure contain the objects from the original structure, and because storage algorithms for different coupling facilities may yield different results, the coupling facility configuration may dictate that additional storage is needed to accommodate the allocation of the new structure. For example, a coupling facility at CFLEVEL=15 would require more storage for a list structure than a coupling facility at CFLEVEL=14 would for the same structure. Conversely, for rebuild purposes, a structure might be allocated with a size smaller than INITSIZE or the value specified or defaulted to for MINSIZE.

When a user-managed rebuild or duplexing rebuild process is initiated, the system determines whether either of the following conditions exist:

- The installation has changed the SIZE specification in the CFRM policy, and the policy change is pending.
- The application has changed any structure attributes on its IXLCONN REBUILD request. (The application may initiate a structure rebuild for the purpose of modifying the structure attributes.)

If either of these conditions is true, the system attempts to allocate the new structure as previously described for initial allocation. If you are migrating to a new coupling facility or a new operating system level when either of these conditions is true, your policy SIZE and/or INITSIZE values must accommodate the required control space; otherwise, the rebuild will fail.

On the other hand, if neither of these conditions exist, or the rebuild process is system-managed, the system allocates the new structure based on the attributes of the initial structure. The system will allocate the structure large enough to contain all the objects that had been allocated in the old structure. The allocated size of the new structure:

- Might be larger than the maximum size indicated by SIZE.
- Might be between the INITSIZE and SIZE values. (Perhaps in order to be able to copy all data that must be copied from the old structure to the new structure.)
- Might be less than INITSIZE. (Perhaps because of a coupling facility storage constraint, but the small size still provided a sufficient number of structure objects to allow the copy process to succeed.)
- Might be less than MINSIZE. (Perhaps because of a coupling facility storage constraint, as long as there is enough space in the new structure to copy all the inuse objects from the old structure.)

After the rebuild process is complete, the installation should display the actual amount of storage allocated to the structure by issuing the DISPLAY XCF,STRUCTURE command. It might then be necessary to update the CFRM policy with appropriate size values so that future initial allocations will not fail. Such failures could occur because during initial structure allocation, the system does not allow the size of a structure to exceed the SIZE specified in the active CFRM policy.

Warning about CF structure size growth: structure rebuild/duplexing, and insufficient CF white space storage capacity for recovery

In order to accommodate possible CF structure size growth when moving CF structures between CFs, z/OS generally allocates an original structure instance by size and ratio in accordance with the structure size specified in the active CFRM policy and the object ratios specified by the structure's exploiter. z/OS then generally allocates a rebuild-new or duplex-secondary structure by counts to get the new structure instance to have the exact same object counts that the original structure instance has.

This is done to ensure that the new structure instance has a sufficient number of objects (such as entries, elements) to be usable by the exploiter as a rebuild-new or duplex-secondary structure, and in particular, to ensure that the structure is adequately sized and has sufficient objects even when the CFLEVELs of the two CFs involved are different. The size of the new structure is allowed to float to whatever size is needed to accommodate the required object counts, to ensure that the rebuild or duplexing operation can succeed. This avoids a variety of difficult problems that might otherwise occur if the new structure were to be allocated by size and ratio, or if the new structure were to be allocated with fewer usable objects despite being allocated with the same absolute size as the original structure (for example, due to CF storage allocation algorithm changes between CFLEVELs).

Although you are warned to expect structure size growth when going from one CFLEVEL to another, and that you must plan for it by providing sufficient CF storage in the new CF to accommodate such growth (and also by re-sizing the structures in the CFRM policy), the following effect is more subtle and must also be planned for: sometimes, even when rebuilding or duplexing between two CFs at the same level, the rebuild-new or duplex-secondary structure will be allocated somewhat larger than the original structure, even though it has identical object counts to the original structure.

The reason for the size discrepancy has to do with difficulties in ascertaining and reproducing the exact structure geometry (including that of various internal structure controls) of the original structure so that it can be replicated in the new structure, even though the object counts of the original structure are known with certainty and can be reproduced exactly. Sometimes, unavoidably, more internal structure controls are allocated for the new structure than were actually present in the original structure, resulting in a larger structure size. This growth does not always occur for every structure, and when it occurs, it typically is on the order of one or two CFCC storage increments (the absolute size of a CFCC storage increment is displayed in the output of the DISPLAY CF command and is documented on a per-CFLEVEL basis in the *PR/SM Planning Guide*).

Note that this growth can be exacerbated in cases where the structure is initially allocated with a given ratio (and its structure controls are allocated based on that ratio), and then the original structure's ratio is significantly modified (with Alter or AutoAlter processing) before the structure getting rebuilt or duplexed. In such a case, the allocation of structure controls in the new structure, which is based on a ratio that is very different from that which was used in allocating the original structure, will potentially use more storage for controls than the original structure used. However, it is still possible for some structure size growth to occur even in structures for which no Alter or AutoAlter processing has ever occurred (and for which such processing may not even be supported).

The system-initiated alter (AutoAlter) algorithm is enhanced to help you avoid reclaims of cache structures by the system when the directory-to-data ratios of the cache structure is high. This enhancement helps AutoAlter to dynamically tune coupling facility placement of cache structures and potentially allows you to avoid coupling facility outages when you are managing CF structure sizes and ratios. Messages IXC347, IXC574I, and IXL015I provide detailed diagnostic information about how the system selects an eligible coupling facility to manage the placement of the structures among a list of preferable coupling facilities. For information about AutoAlter, see *z/OS MVS System Commands*.

Though no CFRM policy size changes need to be made because of this effect, you need to plan your CF storage to provide sufficient CF white space (unused CF storage capacity) to accommodate failure or planned reconfiguration of any given CF, in order to avoid a single point of failure for the sysplex. Since, in such scenarios, structures will need to be rebuilt or undergo duplexing failover, it is imperative that your CF images be sized to have sufficient memory not only to accommodate all of your structures as they are currently allocated, but also how they might become allocated following a rebuild or duplexing rebuild process (including the additional structure size growth described above).

For example, suppose that an installation has two CFs, each at the same CFLEVEL, and each of which has 4 GB of usable space that can be used for structure allocation purposes (after subtracting out the storage that is used for the CF image itself, and is thus unavailable for allocation of structures). Now suppose that a number of CF structures (all simplex) are allocated across the two CF images, and that the total size of all of the allocated structures is exactly 4 GB. This configuration, most likely, is inadequate, despite having 4 GB of allocated structures and 4 GB of CF white space available for recovery of those structures. If one of the CFs fails, the structures in that CF will need to be rebuilt in the surviving CF, and because of the effect described above, the total size of the rebuilt structures will likely be larger than 4 GB. If only 4 GB of space is available, some of the structures might not be able to be rebuilt, leading to sysplex problems and potentially outages.

Alter consideration

Installations that have altered the size of a structure to a size different from either SIZE or INITSIZE in the CFRM policy need not necessarily update those values in the policy before attempting to rebuild the structure.

Understanding MINSIZE/INITSIZE/SIZE correlation

When requesting structure size, take the following into consideration.

- INITSIZE, the size at which you want to initially allocate the structure.
- SIZE, the maximum size of the structure.
- MINSIZE, the minimum bound for structure allocation for structure allocation requests at OS/390 Release 10 and higher.
- CFLEVEL, the level of CFCC in the coupling facility in which a structure is to be allocated.

When allocating the structure initially, whether INITSIZE is specified or not, the system attempts to build all control structures that will be required to support the maximum size of the structure. These control structures are built in the control storage allocation of the structure. For that reason, use care when specifying the maximum structure size with the SIZE parameter in the CFRM policy. IBM recommends that the SIZE value be in a range of about 1.5 to 2.0 times the INITSIZE value. A SIZE value that is greater than twice the INITSIZE might cause the following:

- It might be impossible to allocate a structure at a size of INITSIZE, because the amount of control storage that is required to support the SIZE value might actually be larger than INITSIZE.
- If the allocation succeeds, it might result in a structure with a proportionally large amount of its storage allotted to structure controls, leaving too few structure objects to be exploited usefully by the associated application.

For example, if you have requested a maximum size that is very much larger than the initial size, the system will attempt to use a proportionally large amount of the allocated storage for its controls. The result could be that the allocated storage contains control structures for the future maximum size of the structure and insufficient storage might remain for the application's initial use.

The size of the control structures is affected by the CFLEVEL of the coupling facility in which the structure is to be allocated. Different CFLEVELs will have different control structure requirements, with the result that the same structure could have significantly different sizes depending on the CFLEVEL of the coupling facility. In some cases, a structure might actually become unallocatable in a coupling facility if the control structure allocation was so large that it occupied an amount of space larger than the INITSIZE value. Or, even if the initial allocation succeeds, there might remain in the structure only minimal space for the objects required by the application, thus preventing the application from successfully running. When rebuilding a structure from a coupling facility at one CFLEVEL to a coupling facility at another CFLEVEL, the amount of storage required for the control structures might increase to the point where the structure rebuild would fail because the structure is unallocatable. Increasing the INITSIZE value would be necessary in such instances.

MINSIZE is used primarily to limit the size to which a structure can be altered. However, MINSIZE also has implications at structure allocation time. MINSIZE serves as a minimum bound for the structure size, including the structure allocations during rebuild processing. The only exception to this is during system-managed rebuild processing, when the system will allow the allocation of the rebuild new structure with a size less than MINSIZE as long as there is enough space in the new structure to copy all the inuse objects from the old structure.

Specifying SIZE to be larger than INITSIZE provides flexibility when you need to dynamically expand the size of your structure. However, overspecifying SIZE or not taking into account the CFLEVEL of the coupling facility can result in the allocation of a structure with little usable storage for application data or the inability to allocate a structure at all.

See topic [“Allowing a structure to be altered automatically”](#) on page 63 for more information about system-initiated alter processing. See topic [“CFRM parameters for administrative data utility”](#) on page 346 for more information about specifying structure sizes in the CFRM policy.

Requesting the use of storage-class memory

Storage-class (*flash*) memory provides an overflow capability to minimize the probability of structure-full conditions. Use the CFRM policy SCMMAXSIZE and SCMALGORITHM keywords to enable the use of storage-class memory for a structure.

Note: Coupling facilities defined on the z17 processor or later (CFLEVEL 26 and above) no longer support storage-class memory. Refer to application-specific documentation for alternative means of providing structure overflow capacity.

When the number of structure objects reaches a system-determined threshold percentage of the maximum number that the structure's real storage can contain, a suitably configured coupling facility can *spill* the excess structure objects into storage-class memory. This protects against classes of application failures that cause the rate of writes to the structure to exceed the rate of deletions. However, this protection comes at the expense of potentially degraded performance, as writing to and reading from storage-class memory is slower than accessing the coupling facility's real storage. Coupling facilities at CFLEVELs 19-25 can use storage-class memory.

The use of storage-class memory is not a guarantee that structure-full conditions will not occur, since the storage-class memory itself might fill up. Moreover, not all structure objects are eligible for migration to storage-class memory, and use of storage-class memory requires additional CF real storage (*augmented space*) that might not be available when needed.

Storage-class memory is not allocated to the structure until required for use. Therefore, it is possible to overcommit the storage-class memory that is configured to the coupling facility. It is possible to define the CFRM policy such that the sum of the SCMMAXSIZE values for allocated structures exceeds the total amount of storage-class memory that is available to the coupling facility.

Note: Do not overcommit storage-class memory. This ensures that you have the wanted amount of storage available if an application failure causes the structure to fill up.

The maximum amount of storage-class memory that is available for use with a structure that is allocated in accordance with the SCMMAXSIZE specification may be smaller or larger than SCMMAXSIZE. Various factors can cause the coupling facility to assign a lower maximum value than specified. If there are no limiting factors and the specified size is not an even multiple of the storage-class memory increment, the coupling facility rounds SCMMAXSIZE up to the next highest increment. To display the actual amount of storage-class memory that is assignable to and currently allocated to a structure, issue the **DISPLAY XCF,STRUCTURE** command.

Using storage-class memory requires that the coupling facility allocate augmented space to allow the coupling facility to track the location of data and controls contained in storage-class memory. A small amount of augmented space, referred to as *fixed* augmented space, is always assigned to a structure that is allocated to support storage-class memory. Additional augmented space is allocated from the coupling facility's free space on an on-demand basis, and like storage-class memory itself, is returned to the free pool when no longer required. Neither the fixed nor the dynamic augmented space is included in the reported structure size. When evaluating coupling facility capacity and *white space*, you must consider the amount of augmented space that is required to support the maximum amount of storage-class memory that the facility is potentially using at any one time.

A structure's SCMMAXSIZE specification interacts with its SIZE, INITSIZE, and MINSIZE specifications in a complex manner. The following effects of using storage-class memory are possible:

- The structure size required for a given workload increases significantly, apart from the requirements for fixed and dynamically allocated augmented space. The coupling facility rejects structure allocation requests that specify a structure size that is too small for effective use of storage-class memory.
- The system might initially allocate the structure at a size smaller than MINSIZE.
- The system might report structure-full conditions before either the coupling facility real storage that is allocated to the structure or the maximum specified storage-class memory is full. The coupling facility might reserve some amount of allocated storage to ensure that it has the resources necessary to transfer structure objects between storage-class memory and real storage, and therefore might not make the entire allocated structure capacity available for application requests.

Use the IBM Z Coupling Facility Structure Sizer (CFSizer) at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062) to determine a consistent set of SIZE, INITSIZE, and SCMMAXSIZE specifications that provide the wanted structure capacity.

The maximum amount of storage-class memory that a structure can use is the smaller of the CFRM policy SCMMAXSIZE value and the total amount of storage-class memory that is configured to the coupling facility. (Other factors might even further limit this value.)

The use of storage-class memory increases the amount of control storage that is required to support a given number of entries and elements. Therefore, the use reduces the number of entries and elements that coupling facility real storage can accommodate by a structure of a specified size. If the amount of storage-class memory that is configured to the coupling facility is less than the SCMMAXSIZE specification, and if the structure is therefore allocated to support a smaller maximum amount of storage-class memory than intended by the CFRM policy, it would be capable of containing more entries and elements than it would if allocated in a coupling facility with more configured storage-class memory. That would cause a problem if the structure were to be then rebuilt into a coupling facility with more storage-class memory because the rebuild new structure instance would not be able to accommodate the same object counts as it might when the structure was originally allocated. To prevent this conflict, the system limits the number of entries and elements when allocating in a coupling facility with less storage-class memory than specified by the policy.

To mitigate the performance penalty that is associated with the use of storage-class memory, the coupling facility attempts to migrate to storage-class memory those structure objects that are not expected to be used soon (*pre-staging*). It also retrieves from storage-class memory ahead of time those objects that are expected to be used soon (*pre-fetching*). The migration and retrieval decisions for a structure depend heavily on the associated application's anticipated pattern of structure reference. Therefore, only certain structures are eligible to use storage-class memory. See [Table 30 on page 351](#) for a list of eligible structures and the associated pre-staging and pre-fetching algorithm (SCMALGORITHM) type.

Note: Specification of SCMMAXSIZE and the related keywords for other structures might have unpredictable results, potentially including connect failures or performance degradation.

The following system-determined high and low thresholds control pre-staging and pre-fetching activities into and out of storage-class memory:

- When the number of in-use structure objects of a given type increases above the high threshold percentage of the total number of that object type that can reside within coupling facility real storage, the coupling facility begins migrating structure objects from coupling facility real storage into storage-class memory.
- When the number of in-use structure objects of all migratable types falls below the low threshold percentage of the total number that can reside within coupling facility real storage, the coupling facility begins migrating structure objects from storage-class memory back into coupling facility real storage.

Structure alteration is not permitted while structure objects exist in storage-class memory, or when augmented space other than the fixed augmented space is in use.

CFRM policy changes to the SCMMAXSIZE or SCMALGORITHM keywords cause the policy change to remain pending until the affected structure is deallocated or rebuilt.

Understanding preference and exclusion lists

The coupling facility in which a structure is to be allocated is influenced by preference and exclusion lists that are specified in the CFRM policy for each structure. Each list can have up to eight entries.

- The preference list is an ordered list of **coupling facility** names in which the structure can be allocated.
- The exclusion list is an unordered list of **structure** names with which the structure being defined should not share a coupling facility.

The preference list is used to designate the location of coupling facility structures for performance and capacity considerations. The system will not allocate a structure in a coupling facility that is not listed in its preference list.

The exclusion list can be used to request that related coupling facility structures occupy different coupling facilities if possible. This would prevent a single point of failure for the authorized applications that depend on the use of the related coupling facility structures. The system might allocate a structure in the same coupling facility as a structure in its exclusion list if no other suitable coupling facility can be found.

Developing preference and exclusion lists

When determining which coupling facilities to include in the preference list and which structures to include in the exclusion list, consider various factors. For example,

- If a structure has a backup structure containing the same information, you would want to place the backup structure in a different coupling facility. Specify each structure in the other's exclusion list and specify different preference lists for each structure.
- If two structures with high activity reside in the same coupling facility, you would probably want to separate them for performance reasons. Specify each structure in the other's exclusion list and specify different preference lists for each structure.
- If a structure requires a certain operational level (CFLEVEL) of coupling facility because of the function it provides (such as the structure alter function), you would need to include coupling facilities of only that CFLEVEL in or higher in the structure's preference list.

- If a structure has been defined so that it can be rebuilt (due to coupling facility failure, structure failure, loss of coupling facility connectivity, or for planned coupling facility reconfiguration or maintenance activities), you must list more than one coupling facility in that structure's preference list to allow the structure to be rebuilt in a different coupling facility. The system will not allocate a structure in any coupling facility that is not listed in the structure's preference list.
- If you intend to allow a structure to use storage-class memory, consider including in its preference list only those coupling facilities that are at CFLEVELs between 19 and 25 inclusive and which have storage-class memory configured.
- The site where the coupling facility will reside along with the structure's duplexing options is a factor. For example:
 - A particular simplex structure should be allocated in one site or the other for locality relative to a workload and to avoid remote access latencies unless there is no other alternative. CFRM policy information describing the structure with the workload in SITE1 might be accomplished with:

```
ENFORCEORDER(NO)
DUPLEX(DISABLED)
PREFLIST(CF1SITE1,CF2SITE1,CF1SITE2,CF2SITE2)
```

- A particular duplex structure should be duplexed across sites with the primary instance in a particular site for locality relative to a workload and the secondary instance in the other site unless there is no alternative to duplexing it within the same site. CFRM policy information describing the structure with the workload in SITE1 might be accomplished with:

```
ENFORCEORDER(NO)
DUPLEX(ENABLED,CROSSSITE)
PREFLIST(CF1SITE1,CF2SITE1,CF1SITE2,CF2SITE2)
```

- A particular duplex structure should be duplexed in the same site unless there is no alternative. CFRM policy information describing the structure might be accomplished with:

```
ENFORCEORDER(NO)
DUPLEX(ENABLED,SAMESITE)
PREFLIST(CF1SITE1,CF2SITE1,CF1SITE2,CF2SITE2)
```

- A particular duplex structure must be duplexed in the same site. CFRM policy information describing the structure might be accomplished with:

```
DUPLEX(ENABLED,SAMESITEONLY)
PREFLIST(CF1SITE1,CF2SITE1,CF1SITE2,CF2SITE2)
```

You can specify when formatting your CFRM policy whether the order in which the coupling facilities are listed in the preference list is to be enforced with ENFORCEORDER (YES|NO). Specifying ENFORCEORDER(YES) prevents MVS from reordering the list when attempting to choose the coupling facility in which to allocate a structure. (See “How MVS uses the lists” on page 58.) When listing the coupling facilities, keep in mind that even though some connectivity requirements may have been specified by the application, the system will ignore them if ENFORCEORDER(YES) is specified. For example,

- The CFLEVEL specification will not be honored. This might result in application connect failures if the structure cannot be allocated in a coupling facility of a high enough level.
- The CONNECTIVITY=BESTGLOBAL request also is not honored. This parameter, by definition, specifies that the system is to use the coupling facility selection algorithm.

With ENFORCEORDER(YES), the amount of storage requested for a structure (with the INITSIZE and SIZE CFRM policy parameters) will be honored only to the extent of the available storage in the coupling facility. The system will not choose the next coupling facility in the preference list simply because the first coupling facility did not have the requested amount of storage available. If the system was able to allocate the structure with viable object counts in the first coupling facility in the preference list, then despite the degraded size of the structure, that is where it will be allocated.

How MVS uses the lists

Unless the CFRM policy specifies that the order of coupling facilities in the preference list is to be enforced (ENFORCEORDER=YES), MVS uses the following method of choosing a coupling facility in which to allocate a structure. The system finds the coupling facility identified in the preference list to which the system is connected that best meets the allocation criteria, listed in order of relative importance from most important to least important:

1. For CONNECTIVITY=BESTGLOBAL, coupling facility SFM weight. For a structure without any active connectors, the SFM weight of each coupling facility is the sum of the SFM weights of all systems connected to that coupling facility. For a structure with active connectors, only systems with active connectors are used to determine the SFM weight of the coupling facility. Note that all systems are considered to have equal SFM weight if no SFM policy is active.
2. Connector connectivity to the coupling facility. Note that this only applies to allocation for user-managed rebuild processing that is started with LESSCONNECTION=CONTINUE and causes allocation with CONNECTIVITY=DEFAULT. Otherwise, connector connectivity to the coupling facility is required or already factored into the CONNECTIVITY=BESTGLOBAL SFM weight criteria.
3. Has a coupling facility operational level (CFLEVEL) equal to or greater than the requested CFLEVEL.
4. If this is a duplexing rebuild connect, is isolated from the coupling facility that contains the old structure instance, or is for initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, the facility may provide duplex failure isolation when the structure is subsequently duplexed.
5. Has space available that is greater than or equal to the requested structure size (or has the greatest amount of free space available if any coupling facility does not have enough space for the requested structure size).
6. For duplexing rebuild allocation, the CF satisfies the CROSSSITE DUPLEX preference according to the CFRM active policy for the structure. For initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, the CF may satisfy the preference when the structure is subsequently duplexed.
7. For non-duplexing rebuild allocation of a structure that the system may duplex, the suitability of the CF level of another CF that may be used for duplexing the structure.
8. Coupling facility has enough free space and enough control space to allocate the minimum required structure size.
9. When allocating a Cache structure, the coupling facility has enough free space to allocate the cache structure with "changed data" only.
10. Coupling facility has enough storage-class memory to accommodate in-use structure objects that won't fit in coupling facility real storage.
11. Coupling facility contains enough storage-class memory to allocate the structure with the amount of storage-class memory specified by SCMMAXSIZE.
12. Coupling Facility Operation Level (CFLEVEL) is lower than that specified by the connection or necessary for system-managed processing.
13. Meets the volatility requirement requested. If volatility was requested, is failure isolated from all connectors to the structure. For duplexing rebuild new structure allocation, the volatility requirement is already met when the structure (old instance) is allocated in a non-volatile coupling facility. For initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, the volatility requirement is met if the volatility requirement should be satisfied when the structure is subsequently duplexed.
14. Is a stand-alone coupling facility and therefore failure isolated from all MVS systems. For initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, this occurs when the failure isolation requirement should be satisfied when the structure is subsequently duplexed.
15. Meets the failure-independence requirement requested by the connector. For duplexing rebuild new structure allocation, the failure-independence requirement is already met when the structure (old instance) is allocated in a coupling facility that satisfies the failure-independence requirement. For

initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, the failure-independence requirement is met if the failure-independence requirement should be satisfied when the structure is subsequently duplexed.

16. Does not contain an instance of a structure in the exclusion list of this structure or, if no such coupling facilities exist, does not contain a simplex structure from the exclusion list of this structure.
17. For duplexing rebuild allocation, the CF satisfies the **SAMESITE** or **SAMESITEONLY DUPLEX** preference according to the CFRM active policy for the structure. For initial allocation or non-duplexing rebuild allocation of a structure that the system may duplex, the CF may satisfy the preference when the structure is subsequently duplexed.
18. If system-managed duplexing rebuild is a possibility for this structure, is connected to a remote coupling facility that is also in the **PREFLIST** with sufficient storage for the allocation of a secondary structure in the future.
19. If system-managed duplexing rebuild is a possibility for this structure, is connected to a remote coupling facility that is also in the **PREFLIST**.
20. For **CONNECTIVITY=DEFAULT**, has a higher SFM weight than other coupling facilities with equal criteria.
21. Is located higher on the structure's preference list in the CFRM policy.

If there is no coupling facility in the preference list that meets **all** these criteria, then the system determines the coupling facility that **most closely** meets the criteria. To do this, the system uses a weighting system for each of the coupling facilities in the structure's preference list. The weights correspond to the list of criteria — with system connectivity having the highest weight, **CFLEVEL** the next higher weight, and so on down the list.

Once it has ordered the coupling facilities in accordance with the relative importance of the allocation criteria, the system then attempts to allocate the structure in the coupling facility with the highest weight. If two or more coupling facilities are assigned identical weights, then the selection is made based on the order in which the coupling facilities are defined in the CFRM policy preference list. If the attempt to allocate the structure is not successful, the system attempts structure allocation in successively lower-weighted coupling facilities until allocation is successful. The system will choose the coupling facility that **most closely** meets the requirements of the connect request.

Depending on the coupling facility selected, once the structure is successfully allocated, the structure attributes might not be as requested.

Messages **IXC574I**, **IXC347I**, and **IXL015I** provide an indication of the allocation criteria used to sort the preference list.

If **ENFORCEORDER(YES)** is specified, the system will not reorder the coupling facilities in the preference list, but will attempt to allocate the structure in the exact order specified in the preference list, regulated by the **SAMESITEONLY** duplex site preference.

Specifying a rebuild threshold

If applicable, you can specify, for each structure in the CFRM policy, a value that MVS will use to determine whether to initiate a user-managed rebuild if connectivity is lost to a coupling facility containing that structure. Loss of connectivity can occur because of a failure of a coupling facility attachment or because of certain types of failures of the coupling facility itself.

The appropriate action for MVS to take depends on the scope of the failure and the nature of the sysplex. For example, in a configuration made up of large systems, it might be appropriate to rebuild the structure in another coupling facility so that all of the systems can continue accessing the structure. On the other hand, in a configuration of many small systems, it might be better to simply let one or more of the systems discontinue accessing the structure while the other systems continue their work.

To allow MVS to initiate a user-managed rebuild:

1. Have a structure for which all active connections support user-managed rebuild.

2. Specify a REBUILDPERCENT value in your CFRM policy for each structure that MVS is to evaluate for rebuild, or allow it to default to 1.
3. Optionally, have in place an active SFM policy that supports the use of system weight values for performing recovery actions in the event of loss of connectivity between systems.

Whether MVS initiates a rebuild depends on the application using the structure.

- If the application does not support user-managed rebuild, then even though you specify a REBUILDPERCENT value, the application would not be able to honor your request.
- If the application does support user-managed rebuild, the application can also set its own criteria for when a rebuild is to be started (thus overriding the REBUILDPERCENT value specified in the policy).

How MVS decides to initiate a rebuild

When MVS detects a loss of connectivity, MVS determines the viability of rebuilding each structure affected by the connectivity loss.

If MVS determines that there is an active SFM policy in the sysplex:

1. MVS verifies that the SFM policy data is at the same level on all systems.
2. MVS checks the active CFRM policy to see if a rebuild percent value has been specified for affected structures, or takes the default rebuild percent value of 1.
3. MVS calculates the total system weight of (A) all systems containing at least one active connection to the structure in the coupling facility to which connectivity has been lost, and (B) all systems having at least one active connection to the structure. Note that if there are multiple users of a structure on one system, that system weight is counted only once.

For example, if a structure has one connection per system and all systems are of equal weight 10, then in an eight-system sysplex if one system lost connectivity the value of A (total system weight of all systems containing an active connection that have lost connectivity) is 10 and the value of B (total system weight of all systems containing an active connection) is 80.

4. MVS determines what action is to be taken and informs the connected users through event exit processing.

The determination is arrived at by dividing A by B, multiplying by 100, and then comparing the result with the rebuild percent for the structure in the CFRM policy.

- If the result is greater than or equal to REBUILDPERCENT, then MVS initiates a user-managed rebuild.
- If the result is less than REBUILDPERCENT, MVS does not initiate a rebuild.

In this example, $(10/80)*100$ would be the value compared to the REBUILDPERCENT value. If the value of REBUILDPERCENT was 13 or higher, a rebuild would not be initiated.

If MVS determines that there is not an active SFM policy in the sysplex, MVS initiates user-managed rebuild for loss of connectivity to a structure, regardless of the REBUILDPERCENT specification.

Monitoring structure utilization

One type of monitoring used by the system in a Parallel Sysplex environment is structure full monitoring, which is described here. See [“Duplexing rebuild monitoring” on page 74](#) for a description of the monitoring done by the system for structures that have been enabled by the installation to be duplexed.

Structure full monitoring adds support for the monitoring of objects within a coupling facility structure. Its objective is to determine the level of usage for objects that are monitored within a coupling facility, and issue a warning message if a structure full condition is imminent. This will allow an installation to intervene, either manually or through automation, so that the appropriate diagnostic or tuning actions can be taken to avoid a structure full condition.

Structure full monitoring, running on a given system, will periodically retrieve structure statistics for each of the active structure instances from each of the coupling facilities which it is currently monitoring. The retrieved information will indicate the in-use and total object counts for the various monitored objects. These counts will be used to calculate a percent full value. When structure full monitoring observes that a

structure's percent full value is at or above a percent full threshold in terms of any of the structure objects that it contains, highlighted message IXC585E will be issued to the console and to the system message logs. You can review this message manually and take whatever action is necessary, such as adjusting the structure size or making changes in the workload which is being sent to the structure, or you can define message automation procedures to diagnose or relieve the structure full condition.

For each structure in the CFRM policy, the percent full threshold can be specified by the installation to be any percent value between 0 and 100. Specifying a threshold value of 0 means that no structure full monitoring will take place. If no threshold value is specified, the default value of 80% is used as the full threshold percent value.

When the utilization of all monitored structure objects comes down below the structure full threshold, message IXC586I will be issued to the console and to the system message logs to indicate that the full condition was relieved. Message IXC585E will be deleted.

Only “active” structure instances will be monitored by structure full monitoring. Active structures include:

- Simplex allocated structures.
- Duplexed allocated structures. A duplexed allocated structure will have two allocated instances, a duplexing rebuild old instance and a duplexing rebuild new instance.
- Structures which are in the process of being rebuilt, either as a result of a user-managed or system-managed rebuild process. In this case, the structure will have two allocated instances, a rebuild old and a rebuild new instance.

Active structure instances will be eligible to be monitored by structure full monitoring regardless of whether or not they have any active, or failed-persistent, connectors at the time. Persistent structures with no connectors, or structures which have only failed-persistent connectors, will therefore be monitored.

Structure full monitoring will not monitor “inactive” structures. Inactive structure include structures which are:

- In transition during connect or disconnect
- Pending deallocation as a result of a FORCE structure request that cannot be processed immediately
- Pending deallocation where the structure is being kept because of an associated structure dump which has not been written to a dump data set yet.

In general, structure full monitoring will be discontinued for a structure in all cases where a structure instance transitions from an “active” to “inactive” state, or where the structure instance is deallocated. If structure full monitoring is discontinued for a structure instance for any reason while there is an outstanding IXC585E message, highlighted message IXC587I is issued to the console and to the system message logs. Message IXC585E is deleted.

For Lock structures with record data, an application can reserve a percentage of the total number of record data entries that are to remain free (reserved) after completion of a request that attempts to create a record data entry by specifying a non-zero PCTENTRYRSV value on IXLCONN. See [IXLCONN - Connect to a coupling facility structure](#) in *z/OS MVS Programming: Sysplex Services Reference* for more information. When monitoring a lock structure with reserved record data entries, structure full monitoring will only consider record data entries that are not reserved when determining the percentage full value for the record data entries.

In most cases, structure full monitoring will accurately represent the utilization of objects within the structure being monitored. There are a few special considerations. In the case of a cache structure, you can get a full condition that structure full monitoring is not able to detect, as in the following situations:

- Structure full monitoring does not monitor unchanged objects in a structure, so a cache structure will be monitored only in terms of the changed or locked-for-castout objects that the structure contains. However, a cache structure may be full because all objects are in-use for unchanged objects, in which case structure full monitoring will not detect the full condition. This type of “full” condition will normally cause reclaiming of resources, not a structure full condition.

- A directory only cache structure will not be monitored because it has no changed or locked-for-castout objects for structure full monitoring to count.
- Structure full monitoring will not be able to monitor cache structures allocated on a CFLEVEL=0 coupling facility. Some of the required object counts are not available from this level coupling facility, so it will appear to structure full monitoring that these counts are zero.
- A cache structure with multiple storage classes may experience a “full” condition because its available (unchanged and unlocked) objects are in a storage class where they are not available for satisfying a particular write request. Structure full monitoring will not be able to detect this kind of full condition. It is the responsibility of the structure user to reapportion resources across multiple storage classes via the use of cache reclaiming vectors.

For some structures, being over the structure full monitoring percentage may be a normal condition that does not require any diagnostic or corrective actions. Some examples of situations where structures may go over the structure full percentage without posing a problem are as follows:

- An application may “format” all of the structure objects within a structure before using them. This formatting makes it appear to structure full monitoring that 100% of the structure is in use, causing it to issue message IXC585E, even though there may be plenty of usable space in the formatted structure objects. An example of this kind of protocol is the coupling facility list structure JES2 uses for its checkpoint data. For a structure like this, we would recommend that structure full monitoring be suppressed by setting a FULLTHRESHOLD value of 0.
- Some applications define their own “thresholds” for the structures they use. When the data in these structures exceeds the application-defined thresholds, data gets offloaded, or castout, to DASD. During this process of accumulating data and offloading or casting out, these structures may exceed the structure full monitoring percentage if the installation has defined a higher application threshold for that structure. Message IXC585E will be issued even though the structures have not exceeded the application offload or castout thresholds. Some examples of this kind of protocol are DB2 group buffer pools and logger structures. For a structure like this, we would recommend that the structure's FULLTHRESHOLD value be defined at some value higher than the offload or castout threshold is defined at.

The fact that these structures are over the structure full monitoring threshold does not necessarily indicate that there is a problem. You may choose to suppress or just ignore structure full monitoring messages for these structures.

Monitoring of coupling facilities by structure full monitoring is shared dynamically among systems in the sysplex. The ownership of monitoring by systems can change as systems come and go, or as losses of connectivity to the monitored coupling facilities occur. As a result, you will not be able to use automation that relies on any affinity between a system and the coupling facility being monitored.

Each individual coupling facility is monitored by a particular system, and each system can, in general, monitor zero or more different coupling facilities at the same time.

The monitoring frequency at which structure full monitoring will retrieve structure statistics, running on a given system, from coupling facilities for which it currently owns monitoring responsibilities, is initially set to approximately once every 60 seconds. Structure full monitoring will increase the monitoring frequency to approximately once every 30 seconds if any of the following conditions are met:

- Monitored structures exist that are over full threshold in coupling facilities for which the system currently owns monitoring responsibilities.
- A system-initiated alter is attempted for any eligible structure in coupling facilities which the system currently owns monitoring responsibilities. See [“Allowing a structure to be altered automatically” on page 63](#) for a description of the system-initiated alter processing.

The monitoring frequency will decrease back to approximately once every 60 seconds when all of the following conditions are satisfied:

- All monitored structures are under full threshold in coupling facilities for which the system currently owns monitoring responsibilities.

- A system-initiated alter is not attempted for any eligible structure in coupling facilities which the system currently owns monitoring responsibilities.

Because this monitoring frequency is dynamically tuned, it cannot be relied upon by automation products.

Allowing a structure to be altered automatically

You can specify whether you want the system to automatically alter a structure when it reaches an installation-defined or defaulted-to percent full threshold as determined by structure full monitoring. The alter process may increase the size of the structure, reapportion the objects within the structure, or both. For a structure to be eligible to automatically be altered the following is assumed:

- The application connecting to the structure has specified that alter is allowed by specifying `AllowAlter=YES` on one or more of the `IXLCONN` invocations.
- The installation has specified that system-initiated alter of the structure is allowed (`ALLOWAUTOALT=YES` in the active CFRM policy) and is being monitored by the structure full monitoring function.
- The structure is not in the process of being rebuilt. A structure that is in the process of being rebuilt, either as a result of user-managed or system-managed rebuild, is not eligible to be altered automatically. The exception to this rule is for structures in a user-managed duplexing rebuild environment and structures in the duplex established phase.
- The `SETXCF MODIFY` command was NOT used to disable alter for the structure. Starting CF structure alter processing for such structures is not permitted.

Updating the CFRM policy

For each structure that is to be eligible for system-initiated alter processing, the following parameters in the CFRM policy can be specified:

- `ALLOWAUTOALT`

Specify `ALLOWAUTOALT=YES` to request that the structure be eligible for system-initiated alter processing. For structure alter processing to be started for a structure, alter must be permitted (see the `SETXCF MODIFY` command) and the exploiter must also allow alter. The `ALLOWAUTOALT` specification affects the default value for `MINSIZE`.

The `ALLOWAUTOALT` specification takes effect immediately when the updated CFRM policy is activated.

`ALLOWAUTOALT(NO)` is the default value.

- `FULLTHRESHOLD`

The value you assign to the `FULLTHRESHOLD` parameter is used by both Structure Full Monitoring and system-initiated alter processing. The value is specified as a percentage and represents a percent full threshold in terms of coupling facility structure objects. See [“Monitoring structure utilization” on page 60](#). If a value of zero is assigned to `FULLTHRESHOLD`, neither structure full monitoring nor automatic altering of the structure will occur. If a value other than zero is specified or defaulted to, the value is used as the percent full threshold for structure full monitoring as well as being used by system-initiated alter processing to determine how to alter the structure if the exploiter allows alter. If the exploiter does not allow alter, the `ALLOWAUTOALT` specification is ignored and a non-zero value for `FULLTHRESHOLD` is used only by Structure Full Monitoring to alert the installation.

The `FULLTHRESHOLD` specification takes effect immediately when the updated CFRM policy is activated.

80% is the default value for `FULLTHRESHOLD`.

- `MINSIZE`

The value you assign to `MINSIZE` specifies the smallest size to which the structure can ever be altered, either as a result of system-initiated alter or any alter process (operator command or programming interface). This value also serves as a minimum bound for the structure size on all structure allocation requests (including connect and rebuild connect) with the following exception. During system-managed rebuild, new structure allocation may result in the structure being allocated with a size smaller than the

value specified or defaulted to for MINSIZE. See [“How MVS initially allocates the structure”](#) on page 50 for information about structure allocation during system-managed rebuild.

The MINSIZE value must be less than or equal to the INITSIZE value, or less than or equal to the SIZE value if INITSIZE is not specified. If ALLOWAUTOALT(YES) is specified and MINSIZE is not specified, the system will use a value of 75% of INITSIZE, or 75% of SIZE if INITSIZE is not specified, as the MINSIZE value. Otherwise, if ALLOWAUTOALT(NO) is specified or defaulted to and MINSIZE is not specified, the system will use a value of zero for MINSIZE.

Note that in order for the MINSIZE value to take effect in the sysplex, a newly-activated CFRM policy will remain pending until any allocated structures with a MINSIZE value other than zero have been rebuilt to reflect the policy change. (For structures that are not allocated, the policy change can take effect immediately.) For those allocated structures, you are required to either deallocate or reallocate the structure or issue SETXCF START,REBUILD,STRNAME=strname before the policy change takes effect for the structure.

Structure full avoidance

A system-initiated alter for an eligible structure may begin when structure full monitoring determines that a structure contains monitored objects (e.g. entries, elements, or EMCs) that are at or above structure full threshold specified by FULLTHRESHOLD. For cache structures, structure full monitoring will only consider changed or locked-for-castout objects, since unchanged/unlocked objects are eligible for reclaiming, and therefore do not contribute to a possible structure full condition.

For Lock structures with reserved record data entries, structure full monitoring will only consider record data entries that are not reserved.

For all structure types, each object type will be considered independently within that structure. Therefore, a structure can be above the structure full threshold in terms of one structure object type and be under structure full threshold for another structure object type.

After issuing message IXC585E to externalize the structure full threshold condition, the system may issue an alter request against the affected structure (see [“Conditions preventing a system-initiated alter”](#) on page 65 for the list of Conditions that could prevent a system-initiated alter from occurring). The objective of the alter request will be to reapportion the monitored structure objects, expand the size of the structure, or both, so as to attempt to relieve the observed structure object resource shortage. The attempt to relieve the structure object resource shortage will hopefully avoid the occurrence of a structure full condition which could cause structure updates to fail, which could possibly cause the exploiter that depends on the updates to fail or invoke recovery procedures to handle the structure full condition.

Cache structure reclaim avoidance

A system-initiated alter for an eligible cache structure with data elements may also begin in an attempt to avoid reclaims for directory entries. Structure full monitoring assumes directory entry reclaims are imminent when the total in use (changed and unchanged) percentage of directory entries exceeds 95% of the total directory entries. When this threshold has been exceeded and the size of the structure is larger than 20 megabytes, a system-initiated alter may begin. In this case, the structure size is not expanded, but the structure objects may be reapportioned to provide more cache directory entries at the expense of data elements. System-initiated alter processing will only attempt to increase the entry/element ratio to a maximum value of 255:MAXELEMNUM. MAXELEMNUM refers to the IXLCONN MAXELEMNUM attribute of the structure specified by the application instance that allocated the cache structure.

Relieving coupling facility storage constraints

There are instances when the system will automatically contract a structure when the coupling facility as a whole is at or above 90% full and coupling facility resources are not being used productively by the structure. This processing will potentially initiate alter processing against many structures, reclaiming a small amount of storage from each of them, until the coupling facility storage constraint has been relieved.

The structure must have been defined to be eligible for automatic alter (see [“Allowing a structure to be altered automatically”](#) on page 63 for the list of eligibility requirements). When structure full monitoring

determines that all monitored objects for a particular structure are below the structure full threshold value, the system will start an alter request against the structure to contract the structure. The objective of the alter request is to contract 10 percent of the currently allocated structure size for each eligible structure at any one time. The alter processing will not allow the structure to be contracted below the MINSIZE value specified in the active CFRM policy.

Frequency of system-initiated alters

The frequency at which system-initiated alter processing will attempt to alter a structure is dictated by the monitoring frequency at which structure full monitoring will retrieve structure statistics and evaluate the structures. See [“Monitoring structure utilization” on page 60](#).

Conditions preventing a system-initiated alter

System-Initiated alter processing may not start an alter for an eligible structure when any of the following conditions exist:

- A previously initiated Alter (either system-initiated or initiated by other means) is already in progress for the structure or has recently completed (with alter related events still pending delivery to structure exploiters).
- A cache structure's total in use (changed and unchanged) percentage of directory entries exceeds 95%, but the entry/element ratio exceeds a value of 255:1.
- A structure is already at the maximum size defined by the CFRM policy and the entry/element ratio exceeds a value of 255:1..
- A structure is already at the maximum size defined by the CFRM policy and the current utilization of entries and elements does not require a ratio change (the currently allocated ratio of structure objects is in line with the ratio of structure objects currently being used).
- A structure is already at the maximum size defined by the CFRM policy and the target entry/element ratio calculated by the system-initiated alter processing is equal to the ratio requested on the last alter or allocation request, regardless of whether or not that alter process attained its targets.
- A list structure is already at the maximum SIZE defined by the CFRM policy and the target entry/element ratio and the EMC storage percentage calculated by the system-initiated alter processing is equal to the ratio and EMC storage percentage requested on the last alter or allocation request, regardless of whether or not that alter process attained its targets.
- A structure's current utilization of entries and elements does not require a ratio change (the currently allocated ratio of structure objects is in line with the ratio of structure objects currently being used) and there is no free space in the coupling facility into which to expand the structure.
- A coupling facility as a whole is above 90% full and an eligible structure within that coupling facility contains monitored objects that are not above the structure full threshold specified by FULLTHRESHOLD, but one or more of the monitored object types are sufficiently close to the structure full threshold that an alter initiated to contract the structure might push one or more of the monitored objects over the full threshold value.
- A coupling facility as a whole is above 90% full and an eligible structure within that coupling facility contains monitored objects that are not above the structure full threshold specified by FULLTHRESHOLD, but the currently allocated structure size is sufficiently small that an alter initiated to contract the structure might push one or more of the monitored objects over the full threshold value.

See topic [“Requesting structure size” on page 50](#) for more information about requesting the size for structures. See topic [“CFRM parameters for administrative data utility” on page 346](#) for more information about specifying structure sizes in the CFRM policy.

Determining the amount of coupling facility dump space

The coupling facility dump space is an area within the coupling facility that you reserve for dumping structure data. Systems in the sysplex can request that an SVC dump be taken to capture structure data. SVC dump serializes each structure to be dumped to keep the structure's control information from changing. SVC dump uses the coupling facility dump space to quickly capture structure control

information so that dump serialization is held for as brief a time as possible. The captured control information is written to an SVC dump data set after dump serialization is released. (Note: The maximum amount of time that SVC dump is allowed to hold serialization on the structure is specified initially when the application connects to the structure. The operator can override that amount of time with the DUMP or CHNGDUMP command.)

You can define only one dump space in each coupling facility; however, that dump space can contain several structure dump tables. IBM recommends that you specify 5% of the available storage in a coupling facility for dump space. If you do not specify any coupling facility dump space, none will be allocated and no structures in the coupling facility can be dumped.

You can determine how the dump space in a coupling facility is being utilized by using RMF reports or the DISPLAY CF command. Information returned includes the total amount of storage in the coupling facility, the amount not being used, and information about the dump space.

You might need to change to a larger coupling facility dump space if you find that you are receiving partial SVC dumps because there is not enough available space to hold the structure information. Conversely, you might need to decrease the amount of dump space if you find that the space is not being used or that its size is causing structure allocations to fail or to be allocated in a less preferable coupling facility.

To change the size of the dump space defined in the active CFRM policy, you must redefine the amount of dump space for the coupling facility in a CFRM administrative policy and follow normal procedures for switching from one policy to another, such as with the SETXCF START,POLICY command.

Starting a CFRM policy

To start using a CFRM policy in your sysplex, issue the SETXCF START ,POLICY command, specifying the policy name and type(CFRM). The system identifies the policy, which is stored as an administrative CFRM policy on the CFRM couple data set, and makes a copy of it on the same CFRM couple data set as the “active” CFRM policy. The administrative copy of the same policy remains intact on the CFRM couple data set. See [“Two ways to start a CFRM policy” on page 48](#).

The system will add real-time data about sysplex and coupling facility operations to the area of the CFRM couple data set holding the active policy; the administrative copy of the policy will not be changed unless you use the administrative data utility (IXCMIAPU) to change it.

Updating a CFRM policy

The CFRM policy resides on a couple data set that you have previously formatted to hold not only the policy but also status data about the resources defined in that policy. For example, if you formatted a CFRM policy that defined 64 structures, then the system would ensure that enough space was allocated on the couple data set to hold status information for 64 structures.

When updating a CFRM policy, you must determine whether you need to reformat the couple data set or whether you can update the existing policy with new information.

- If the update requires increasing the size of the couple data set in which the policy resides, then you must first format a new couple data set in which the policy is to reside using the IXCL1DSU format utility.
- If the update does not require increasing the size of the couple data set, then you can use the IXCMIAPU administrative data utility to create your updates.

So, for example, to increase the number of structures that can be defined in a policy from 64 to 128, you would have to format a new couple data set for the policy.

Formatting a new CFRM couple data set

“Format Utility for Couple Data Sets” describes the IXCL1DSU format utility and the resources that can be defined in a CFRM policy — number of policies, number of coupling facilities within a policy, number of structures within a policy, and number of connections to a coupling facility structure. If the update to an existing policy involves an increase in any of these resources, you must format a new couple data set.

When formatting the couple data sets to support CFRM policy data, ensure that the value specified for MAXSYSTEM matches the value that was specified for MAXSYSTEM when the sysplex couple data sets were formatted.

Updating an existing CFRM couple data set

To update an administrative policy that you have created (or add a new administrative policy) in the CFRM couple data set, you must run the administrative data utility (IXCMIAPU) against the CFRM couple data set that contains both the active policy and the set of administrative policies. The utility adds or changes policy data in the administrative policies **only**. The utility does not change any information in the system's copy of the active CFRM policy.

For example, if you want to update information in your active CFRM policy, you would run IXCMIAPU to change the administrative copy of that policy. (You omit the DSN and VOLSER keywords in the IXCMIAPU utility when updating administrative policies.) To switch to the updated policy, issue the SETXCF START,POLICY command. This command causes the system to copy your updated policy to the active policy. It is possible that not all changes that you specified in the updated policy will take effect immediately. See [“Transitioning to a new CFRM policy” on page 86](#) for a description of how CFRM updates an active policy.

Another way to update your active policy without losing the ability to return to it later is to make a copy of the policy, apply the necessary definition changes, and rename the policy. For example, if your active policy is POLICY1, define a new POLICY2 with identical definitions except for those that you want to change. If, after you activate the newly created policy, POLICY2, you decide to return to the original policy, POLICY1, you can issue the SETXCF START,POLICY specifying POLICY1.

Summarizing CFRM policy setup requirements

To set up a CFRM policy, you need to:

- Format a CFRM couple data set. Use the IXCL1DSU format utility program. Ensure that the CFRM couple data set is available to appropriate systems in the sysplex that plan to use coupling facility resources.

If XCF signaling is to be accomplished through structures in a coupling facility, the couple data set that is to contain the CFRM policy must be formatted for the same number of systems as the sysplex couple data set. The MAXSYSTEM value used when formatting the CFRM couple data set should match the MAXSYSTEM value used when formatting the sysplex couple data set. If the MAXSYSTEM value for CFRM couple data set is less than that of the sysplex couple data set, systems may be prevented from joining the sysplex because the CFRM couple data set has not been formatted to support the additional systems.

(Chapter 11, “Format utility for couple data sets,” on page 313 describes the IXCL1DSU format utility and gives an example of its use.)

- Define one or more CFRM administrative policies.

The administrative data utility, IXCMIAPU, described in [Chapter 12, “Administrative data utility,” on page 335](#), provides the interface to associate the definitions with a policy name and to place the policy in a pre-formatted CFRM couple data set. IXCMIAPU also is used to update existing administrative policies that exist in the CFRM couple data set.

Definitions include information about each coupling facility and its space allocation and each structure and its preferred location in a particular coupling facility.

- Start one of the defined administrative policies, making it the “active” CFRM policy for the sysplex.

To activate a CFRM policy named POLICY1 that is defined in the CFRM couple data set, issue the following command:

```
SETXCF START,POLICY,POLNAME=POLICY1,TYPE=CFRM
```

If a policy already is active when you issue the START,POLICY command to start a (non-active) administrative policy, the system begins its transition to using the new policy. See [“Transitioning to a new CFRM policy” on page 86](#) for a description of this transition process.

To deactivate a CFRM policy, issue the SETXCF STOP,POLICY,TYPE=CFRM command.

Comparing message-based processing and policy-based processing

IBM recommends that you enable message-based processing because of the performance, availability, and scalability benefits that this protocol can provide. The message-based protocol offers advantages over the policy-based protocol for Parallel Sysplex configurations that have many structures and structure connectors, especially during scenarios that involve system failure and coupling facility recovery.

For example, consider the role of the CFRM couple data set in a recovery situation. With the policy-based mode of CFRM processing, the CFRM couple data set, as the centralized repository for coupling facility resource usage, often becomes a bottleneck when recovery for a failed system or coupling facility is required. This recovery implies recovery for multiple structures and connectors. Each element of recovery requires every surviving system to access the CFRM couple data set under serialization, implying that access is limited to one system at a time. Because each system potentially must access the CFRM couple data set many times, contention on the data set greatly slows the recovery process.

In contrast to the policy-based protocol, the message-based protocol defines a single system as "manager" and all other systems as participants in the recovery process. The manager system is responsible for coordinating events and confirmations with the participant systems and is responsible also for updating the CFRM active policy, thus reducing I/O to the CFRM couple data set to a single, central control point. Communication between the manager system and the participant systems, on which applications or subsystems are running, is through XCF signaling.

Message-based processing applies on a structure basis. XCF signaling structures are not eligible to use message-based processing; these structures use policy-based processing.

With z/OS V2R1, message-based processing is enhanced with the ability to perform CFRM LOSSCONN recovery management. When the optional CFLCRMGMT function is enabled on all active systems in the sysplex, the message-based manager system can enable CFRM LOSSCONN recovery management. CFRM LOSSCONN recovery management can enhance average CF LOSSCONN recovery time by processing CF structures serially, rather than in parallel. Without CFRM LOSSCONN recovery management, systems independently initiate processing (rebuild, for example) to recover from a loss of connectivity to a CF and all that processing is initiated simultaneously. Performing recovery processing for many structures in parallel tends to result in recovery processing for each structure taking a long time. With CFRM LOSSCONN recovery management, the message-based manager system is responsible for initiating processing to recover from a loss of connectivity to a CF and all that processing is initiated in a more serial manner. The serial nature of the recovery processing can help reduce interference and reduce average recovery time. When a system that runs on a z15 server that supports recovery process boosts participates in loss of connectivity recovery processing, the system requests System Recovery Boost's recovery process boost support to further reduce average recovery time. For more information about System Recovery Boost, see [IBM Z System Recovery Boost Content Solution \(www.ibm.com/support/z-content-solutions/system-recovery-boost/\)](http://www.ibm.com/support/z-content-solutions/system-recovery-boost/).

As part of enabling a sysplex for message-based processing, you must provide a version of the CFRM couple data set that is formatted to support message-based processing. When you run the IXCL1DSU format utility, include the keyword ITEM NAME(MSGBASED) to have the CFRM couple data set formatted. For information about formatting the CFRM couple data set, see [“CFRM parameters for format utility” on page 319](#).

When a primary CFRM couple data set that supports message-based processing is switched into use (or used during system initialization), the system automatically switches to message-based processing because that is the preferred mode of processing. Message-based processing is automatically enabled for each structure for which no events are pending delivery for either the structure or connectors to the structure.

You cannot suppress the system triggered switch to message-based processing. However, you can enable or disable message-based processing, nondisruptively, according to your installation's requirements. To enable or disable message-based processing, issue the SETXCF [START|STOP],MSGBASED command.

The **SETXCF** command can be used to change the message-based manager system. To do so, issue the SETXCF STOP,MSGBASED command from any system in the sysplex, then issue the SETXCF

START, MSGBASED command from the system that you want to be the new manager system. During the time that a new manager system is not defined, you lose the performance enhancements of less contention and typically faster processing of CFRM that MSGBASED provides. You can determine at any time which system is the message-based manager system by issuing the DISPLAY XCF, STRUCTURE command.

```
EVENT MANAGEMENT: MESSAGE-BASED MANAGER SYSTEM NAME: sysname
MANAGEMENT LEVEL : 01050107
```

For more information about the **SETXCF** command, see [SETXCF command in z/OS MVS System Commands](#).

Enabling message-based processing does not mean that all structures begin using that protocol immediately. When message-based is enabled sysplex-wide (through the SETXCF command, sysplex initialization, or PSWITCH processing), each individual structure setting remains policy-based until the system can use the message-based protocol for the structure. Specifically, a structure remains policy-based until the following conditions are satisfied:

- Structure is not an XCF signaling structure
- Processing is complete for any events that occurred before message-based processing was enabled for the system
- At least one new event occurred in the time since message-based processing was enabled for the system.

Until these conditions are met, the structure remains policy-based, and is indicated as such by the DISPLAY command. After the conditions are met, the structure begins by using the message-based protocol.

In contrast, if message-based processing is disabled sysplex-wide, all structures are disabled for message-based processing immediately.

An installation guide to duplexing rebuild

Duplexing rebuild is a process by which a duplexed copy of a structure is created and maintained, so that in the event of a failure, a viable structure will remain available to the application. For either method (user-managed or system-managed) of duplexing rebuild, the installation has the final word on whether it will allow a structure to be duplexed by controlling the setting of the DUPLEX keyword in the installation's active CFRM policy.

- User-managed duplexing rebuild is available only for cache structures. Operations to the structure are maintained in a synchronized manner through unique protocols designed by the subsystem. At the current time, it is used by DB2 for its group buffer pools. When an exploiter of duplexing rebuild supports both methods (user-managed and system-managed), the user-managed method will always take precedence over the system-managed method.
- System-managed duplexing rebuild applies to all structure types. Operations to the structure are maintained in a synchronized manner through protocols established by z/OS. Any application that provides support for system-managed processes can participate in either system-managed rebuild or system-managed duplexing rebuild. See [“Coupling facility structures for IBM products”](#) on page 96.
- System-managed asynchronous duplexing rebuild is available only for lock structures. Operations to the structure are asynchronously forwarded by the primary coupling facility to the secondary structure. Any application that provides support for system-managed asynchronous duplexing rebuild provides support for system-managed processes, and therefore also supports system-managed rebuild and system-managed duplexing rebuild. When exploiters support system-managed asynchronous duplexing, the duplexing mode will be determined by the CFRM policy.

Understanding system-managed duplexing rebuild requirements

The installation must play a significant role in the establishment of system-managed duplexing rebuild for one or more of its coupling facility structures. Both hardware and software requirements must be

understood and met, evaluations of structure support for system-managed processes must be made, migration plans must be put into place, and the need for additional system monitoring of hardware resources (both for capacity planning and for performance) must be factored in.

Hardware requirements

Coupling facility: At least two coupling facilities at CFLEVEL=11 or higher are required. IBM recommends a minimum of three coupling facilities at CFLEVEL=11 or higher so that if one coupling facility is removed for maintenance, there will still exist two others to contain the duplexed pair of structures. The coupling facilities must all be defined in the preference list for the structure to be duplexed. You might need to add additional coupling facility storage resources and processor capacity, as well as additional z/OS to CF link capacity, because two structure instances will be allocated for an extended period of time and for each duplexed coupling facility operation, two operations will be driven, one for each structure.

See “[Understanding the coupling facility level \(CFLEVEL\)](#)” on page 42 for sizing and other considerations before migrating to any new CFLEVEL.

- **CFLEVEL=16 to CFLEVEL=16 duplexing:** You can improve duplexing performance by duplexing between two coupling facilities at CFLEVEL=16 or higher. See [Coupling Facility Level \(CFLEVEL\) Considerations \(www.ibm.com/docs/en/systems-hardware/zsystems/9175-ME1?topic=considerations-coupling-facility-level-cflevel\)](#) for information on what server models support CFLEVEL 16. To take advantage of this performance optimization, you must enable the optional DUPLEXCF16 function using the COUPLExx FUNCTIONS statement (see [z/OS MVS Initialization and Tuning Reference](#) for more information) or the SETXCF FUNCTIONS command (see [z/OS MVS System Commands](#) for more information).
- **CFLEVEL=21 to CFLEVEL=21 asynchronous duplexing:** You can improve system-managed duplexing performance for lock structures by exploiting asynchronous duplexing, in which updates to the primary coupling facility structure are asynchronously forwarded to the secondary structure. To enable this performance improvement, you must use the DUPLEX ASYNC or ASYNCONLY keyword in the CFRM policy.
- **CFLEVEL=22 (or higher) duplexing:**
Server models that support CFLEVEL 22 have been enhanced to improve the efficiency of scheduling and processing work targeted to system-managed synchronous duplexed CF structures. To take advantage of this improvement without sacrificing duplexing reliability, you must run z/OS V2R3 or a lower release with a PTF for APAR OA52058 when a CFLEVEL=22 CF is involved in duplexing. See [Coupling Facility Level \(CFLEVEL\) Considerations \(www.ibm.com/docs/en/systems-hardware/zsystems/9175-ME1?topic=considerations-coupling-facility-level-cflevel\)](#) for information on which server models support CFLEVEL 22.

CF-to-CF Links: CF-to-CF links provide connectivity between a pair of coupling facilities in which a duplexed pair of structure instances are to be allocated. The link connectivity between the coupling facilities must be bidirectional, and there should be more than one link providing CF-to-CF connectivity in each direction for highest availability.

Software requirements

- Subsystems that exploit system-managed duplexing rebuild may require new levels in order to provide their support of the function.
- System-managed asynchronous duplexing requires the PTF for APAR OA47796.

Deciding whether to duplex a structure

The decision to enable a particular structure for system-managed duplexing rebuild is done at the installation level, through the DUPLEX keyword in the CFRM policy. Even in a hardware-software configuration that is fully supportive of system-managed duplexing rebuild, it may not be to the installation's benefit to exploit the function for certain of its structures. The installation must consider, on a structure by structure basis, whether the availability benefits to be gained from system-managed duplexing rebuild's robust failure recovery capability for a given structure outweigh the additional

costs associated with system-managed duplexing for that structure. The availability benefit depends on considerations such as whether or not the structure exploiter currently supports rebuild, whether that rebuild process works in all failure scenarios, and how long that rebuild process generally takes, among others, compared to the rapid failover recovery capability that duplexing provides. The costs depend on considerations such as the hardware configuration cost (possible additional coupling facility processor and CF link capacity requirements), software configuration cost (possible upgrades to installed software levels), and the coupling efficiency cost of duplexing the structure given its expected duplexing workload.

Based on these considerations, an installation might well make a conscious decision to duplex only a subset of the structures that support duplexing, given their installation's unique workload, considerations, and tradeoffs. For example, some installations might choose to duplex all structures that do not provide any user-managed rebuild recovery mechanism, but leave all structures that do provide a user-managed recovery capability unduplexed. Other installations might also want to duplex some of the structures that do have a user-managed rebuild capability, because duplexing provides a faster or more robust recovery mechanism than rebuild does under some circumstances. Other installations might choose to duplex all eligible structures except for a few intensively-used data sharing and duplexing intensive structures for which the coupling efficiency cost is not justified in terms of the availability advantage that duplexing provides. And of course, some installations might choose to duplex all eligible structures.

Consider the following for each of your installation's structures when deciding whether to enable it for system-managed duplexing rebuild:

- Does the structure exploiter support system-managed processes?
- Is the correct level of exploiter software installed?
- If user-managed rebuild is supported, is its functionality sufficient to override the need for duplexed structure instances? The structures that benefit most from duplexing are:
 - Those that do not support rebuild
 - Those that support rebuild but cannot rebuild in some scenarios, such as when a connector fails at the same time the structure or coupling facility fails, or
 - Those whose rebuild support is time-consuming or error prone.
- Does the workload involve high rates of write activity to particular structures? (For example, lock and list structures used for logging purposes are “write only” for the most part. All write operations will have to be duplexed.)

Migration steps

To allow system-managed duplexing rebuild in a sysplex, the installation must do the following:

1. Determine which structure exploiters support system-managed duplexing. Evaluate the necessity or the desirability of using the system-managed duplexing rebuild function on a structure-by-structure basis for each subsystem.
2. Migrate at least one system to z/OS V1R2 before making the necessary coupling facility configuration changes. The DISPLAY command in z/OS V1R2 has been enhanced to allow you to display the CF-to-CF link connectivity, thus easing your ability to establish this connectivity.

System-managed duplexing rebuild requires that the CFCC be at CFLEVEL 11 or higher and that CF-to-CF link connectivity be established between any pair of coupling facilities in which duplexed structures are to be allocated. For highest availability, it is recommended that the CF links be redundant, that is, more than one link in each direction. Prior to IBM eServer zSeries 900, separate sender and receiver links between coupling facilities were required. With the zSeries, coupling facilities can be connected with a single bidirectional link. As recommended, more than one of these links should be provided for higher availability.

3. Migrate all systems to z/OS V1R2.
4. Format CFRM couple data sets (primary and alternate) to support system-managed duplexing rebuild (include SMDUPLEX statement in IXCL1DSU). Bring the new couple data sets into use by adding one as a new alternate, PSWITCH to make it the primary, and then bring the second into the configuration as the alternate.

RECOMMENDATION FOR AVOIDING SYSPLEX-WIDE IPL:

Be aware that once these CFRM couple data sets that support system-managed duplexing rebuild are in use, no systems at a release earlier than z/OS V1R2 will be able to use the CFRM function in the sysplex. Also note that if it is necessary to revert or fallback to a CFRM couple data set that does not have the capability of supporting system-managed duplexing rebuild, a sysplex-wide IPL will be required for all systems using the CFRM function. (However, it should not really ever be necessary to fall back from a CFRM couple data set that supports SMDUPLEX to one that does not. An alternative to doing this would simply be to disable duplexing for structures by changing the policy to one that specifies or defaults to DUPLEX(DISABLED) for the appropriate structures, or some desired subset of structures.)

5. Evaluate the effectiveness of enabling the structure for system-managed duplexing rebuild. For each structure that you have decided to use system-managed duplexing rebuild, migrate the exploiting subsystem or product to the level that supports system-managed duplexing rebuild. IBM recommends that you do not define any structures in your CFRM policy as enabled for duplexing until the exploiting subsystem has been upgraded to the duplexing-capable level on all systems in the sysplex. While a structure is duplexed, downlevel connectors that do not support system-managed duplexing rebuild will be unable to connect to the duplexed structure.

Create or modify a CFRM policy with the IXCMIAPU utility to enable the structure for system-managed duplexing rebuild (DUPLEX keyword) and determine the placement of the duplexed structures (preference list). Start the policy to make it the active CFRM policy in the sysplex.

Continue this process on a structure-by-structure basis until all structures intended to be enabled for system-managed duplexing rebuild have been deployed.

The decision to allow or enable system-managed duplexing rebuild for each structure should be made on a subsystem or product basis. The installation can thus ensure that the appropriate software levels are in place to support system-managed duplexing rebuild on all systems, that sufficient coupling facility resources are available to support the duplexing workload for the structure, and that any performance considerations that are associated with duplexing the structure have been evaluated.

After system-managed duplexing rebuild is operational in the sysplex, it is also the responsibility of the installation to monitor structure placement and performance and make any necessary tuning changes.

Refer to [“Migration steps to enable system-managed duplexing of logger structures”](#) on page 288 for migration steps that you must take to enable system-managed duplexing of Logger structures.

Migration steps for system-managed asynchronous duplexing

To allow system-managed duplexing rebuild in a sysplex, the installation must take the following steps:

1. Determine which structure exploiters support system-managed asynchronous duplexing. Evaluate the necessity or the desirability of using the system-managed asynchronous duplexing function on a structure-by-structure basis for each subsystem.
2. Install APAR OA47796 on all z/OS V2R2 systems.
3. Format CFRM couple data sets (primary and alternate) to support system-managed asynchronous duplexing (include the ASYNCDUPLEX statement in IXCL1DSU). Bring the new couple data sets into use by adding one as a new alternate, PSWITCH to make it the primary, and then bring the second into the configuration as the alternate.

RECOMMENDATION FOR AVOIDING A SYSPLEX-WIDE IPL

Once these CFRM couple data sets that support system-managed asynchronous duplexing are in use, no systems at a maintenance level earlier than z/OS V2R2 with APAR OA47796 can use the CFRM function in the sysplex. If it is necessary to revert or fall back to a CFRM couple data set that does not have the capability of supporting system-managed asynchronous duplexing, a sysplex-wide IPL is required for all systems that use the CFRM function. (However, it should never be necessary to fall back from a CFRM couple data set that supports ASYNCDUPLEX to one that does not. An alternative would be to disable asynchronous duplexing for structures by changing the policy to one that specifies or defaults to DUPLEX(DISABLED) or DUPLEX(...,SYNCONLY) for the appropriate structures, or some desired subset of structures.)

4. Evaluate the effectiveness of enabling the structure for system-managed asynchronous duplexing. For each structure that you want to use system-managed asynchronous duplexing, migrate the exploiting subsystem or product to the level that supports system-managed asynchronous duplexing. IBM® recommends that you do not define any structures in your CFRM policy as enabled for asynchronous duplexing until the exploiting subsystem has been upgraded to the asynchronous duplexing-capable level on all systems in the sysplex. While a structure is being asynchronously duplexed, connect requests from downlevel connectors that do not support system-managed asynchronous duplexing will cause duplexing to be stopped.

Use the CF Structure Sizer (CFSizer) tool to determine the appropriate structure sizes for allocating the applicable structures with support for asynchronous duplexing. The CFSizer tool is available at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062).

Create or modify a CFRM policy with the IXCMIAPU utility to enable the structure for system-managed asynchronous duplexing (ASYN or ASYNCONLY parameter for the DUPLEX keyword), provide the new sizes that are determined by CFSizer, and determine the placement of the duplexed structures (preference list). Start the policy to make it the active CFRM policy in the sysplex.

Continue this process on a structure-by-structure basis until all structures intended to be enabled for system-managed asynchronous duplexing have been deployed.

The decision to allow or enable system-managed asynchronous duplexing for each structure should be made on a subsystem or product basis. The installation can thus ensure that the appropriate software levels are in place to support system-managed asynchronous duplexing on all systems, that sufficient coupling facility resources are available to support the duplexing workload for the structure, and that any performance considerations that are associated with asynchronously duplexing the structure have been evaluated.

Once system-managed asynchronous duplexing is operational in the sysplex, it is also the responsibility of the installation to monitor structure placement and performance and make any necessary tuning changes.

Controlling the use of the duplexing rebuild process

Whether it be user-managed or system-managed, the installation controls whether it will allow a structure to be duplexed on a structure-by-structure basis. The DUPLEX keyword in the CFRM active policy applies both to user-managed and system-managed duplexing rebuild. Note that it is possible to initially define a policy that specifies DUPLEX(DISABLED) for a structure and then at a later time bring a new policy into the sysplex that allows or enables duplexing rebuild to be initiated for the structure and vice versa.

Policy changes affecting duplexing rebuild processes

A change in the DUPLEX keyword from DISABLED to either ALLOWED or ENABLED, or vice versa, might become effective immediately. DUPLEX(DISABLED) and the default duplexing mode of SYNCONLY both indicate that a structure will be allocated without the ability to participate in asynchronous duplexing. Other duplexing mode specifications indicate that a structure should be allocated with the ability to participate in asynchronous duplexing. A policy change becomes pending if it changes the asynchronous duplexing indication. Otherwise, a change to the DUPLEX parameter takes effect immediately with policy activation.

Even if a change in the DUPLEX keyword becomes effective immediately, the effect of the change differs between user-managed and system-managed duplexing rebuild if other changes that become pending are also made to the policy at the same time. For user-managed duplexing rebuild, the allocation or deallocation of the duplexed structure causes all pending policy changes to go into effect. For system-managed duplexing rebuild, the process cannot be started or stopped when policy changes are pending. If one or more policy attributes are changed in addition to the DUPLEX attribute, the system is prevented from initiating duplexing rebuild because of the other pending policy changes.

Overview of the system-managed duplexing rebuild process

System-managed duplexing rebuild is a process managed by the system that allows a structure to be maintained as a duplexed pair. The process is controlled by CFRM policy definition as well as by the subsystem or exploiter owning the structure. The process can be initiated by operator command (SETXCF), programming interface (IXLREBLD), or be MVS-initiated. Note that user-managed duplexing rebuild is controlled and initiated in the same manner as system-managed duplexing rebuild but is managed by the subsystem or exploiter owning the structure and applies only to cache structures.

Duplexing rebuild monitoring

When DUPLEX(ENABLED) is specified for a structure in the CFRM active policy, the system will attempt to initiate duplexing (either user-managed or system-managed) for those eligible structures that are not currently duplexed. Note that user-managed duplexing rebuild, which applies only to cache structures, will be given precedence over system-managed duplexing rebuild when the structure exploiter supports both of these processes.

The conditions under which the system will attempt to initiate a duplexing rebuild are:

- Occurrence of a connect to a structure
- Occurrence of a disconnect
- Change in the CFRM policy
- Occurrence of a FORCE to a structure
- System gains connectivity to a coupling facility
- First system in a sysplex gains access to a coupling facility
- Reconciliation (comparison) of a coupling facility's structure contents to a CFRM policy contents

Conditions that apply only to system-managed duplexing rebuild are:

- Gain of CF-to-CF link connectivity
- Issuance of a PSWITCH command to a CFRM couple data set that supports SMDUPLEX

Additionally, when DUPLEX(ENABLED) is specified for a structure in the CFRM active policy that is not duplexed, the system will set up a monitor to periodically check for the following conditions:

- Completion of a rebuild or duplexing rebuild
- Release of structure dump serialization
- Deallocation of a structure
- Completion of structure alter or contraction
- Reduction in the size of the coupling facility dump space

When coupling facility resources become available as a result of one of these events listed above, such that a structure that previously could not be duplexed because of a lack of resources can now be duplexed, the system will eventually (but not immediately) attempt to duplex one or more structures using the newly available resources. The timing is dependent on that of the periodic monitoring.

When any of the conditions listed occurs, including those for which the system is periodically monitoring, the system will evaluate the feasibility of starting a duplexing rebuild of the appropriate type (user-managed or system-managed). If it appears that a duplexing rebuild cannot be initiated for a structure, given current conditions, the system will issue the following messages:

- Message IXC574I is issued to the hardcopy log to document why starting a duplexing rebuild was not deemed feasible.
- Message IXC538I is issued to the console to indicate the reason why the duplexing rebuild process could not be initiated. This message might be issued repeatedly due to the system's periodic monitoring of structures that specify DUPLEX(ENABLED) in the CFRM active policy. The installation might need to consider making configuration changes or taking other actions in order to allow duplexing to be started for the structure. It might also be appropriate to change the CFRM policy from DUPLEX(ENABLED) to either DISABLED or ALLOWED until the configuration changes can be made.

MVS will stop a duplexing rebuild process under the following conditions:

- For user-managed duplexing rebuild, if the CFRM policy is changed such that one of the coupling facilities in which the duplexed pair of structures is allocated is removed from the preference list, the duplexing rebuild is stopped.
- For both user-managed and system-managed duplexing rebuild, if the CFRM policy for DUPLEX is changed to DISABLED, the duplexing rebuild is stopped. Other policy changes such as preference list, structure size, etc. will not result in system-managed duplexing rebuild being stopped, but may result in a change becoming pending against the duplexed structure.

Once a duplexing rebuild is stopped, the following differences exist between user-managed and system-managed duplexing rebuild:

- For user-managed duplexing rebuild, pending policy changes do not prevent user-managed duplexing rebuild from starting. After the stop process completes, user-managed duplexing rebuild will be started again and the pending policy changes will take effect when the new structure is allocated.
- For system-managed duplexing rebuild, pending policy changes prevent system-managed duplexing rebuild from starting. Only a change to the DUPLEX setting takes effect immediately; if other changes cause the CFRM policy to become pending, system-managed duplexing rebuild cannot be started until the pending policy change for the structure takes effect.

Selecting a coupling facility for the duplexed pair of structures

• Allocating the Primary Structure

Prior to initial allocation of a structure, the system is aware of the structure's candidacy for system-managed duplexing rebuild. Therefore, for those structures that might eventually be duplexed, the system uses a modified allocation algorithm that gives preference to allocating the original structure in a coupling facility of CFLEVEL=11 or higher, that has CF-to-CF connectivity to at least one other coupling facility of CFLEVEL=11 or higher, which has sufficient space to contain a secondary instance of the structure and is in the preference list for the structure. In the event that a coupling facility meeting the criteria cannot be found, the system selects a coupling facility for structure allocation in the following order of preference:

- A CFLEVEL=11 or higher coupling facility that has CF-to-CF link connectivity to another CFLEVEL=11 or higher coupling facility in the preference list which **does not currently have** sufficient available space to contain the secondary instance of the structure.
- A CFLEVEL=11 or higher coupling facility that does not have CF-to-CF link connectivity to another CFLEVEL=11 or higher coupling facility in the preference list.
- A coupling facility below CFLEVEL=11.

• Allocating the Secondary Structure

The system guarantees that the primary and secondary structures get allocated in different coupling facilities from one another and gives strong preference to placing the primary and secondary structures in two different coupling facilities that are failure-isolated with respect to each other. If the secondary structure cannot be allocated in a particular coupling facility, message IXC574I indicates the reason.

The allocation algorithm for the secondary structure takes the following considerations into account:

- ENFORCEORDER

When the installation has specified in the CFRM policy that the preference list order is to be strictly enforced for the structure, the system will only apply those eligibility list considerations (listed below) that involve dropping ineligible coupling facilities from the preference list. MVS will not apply any of the considerations that involve weighting the coupling facilities and reordering the preference list based on these attribute weights (considerations for volatility, failure-isolation from connectors, failure-isolation from primary coupling facility, and exclusion list).

- CF-to-CF Link Connectivity

At the time of the secondary structure allocation, there must be CF-to-CF link connectivity between the coupling facility in which the primary structure is allocated and any coupling facility in which the secondary structure is to be allocated. Any coupling facility in the preference list that does not have CF-to-CF link connectivity to the coupling facility where the primary structure resides is dropped from the eligibility list.

- LOCATION(OTHER)

The secondary structure cannot be allocated in the same coupling facility as the primary structure under any conditions. The coupling facility that contains the primary structure is dropped from the eligibility list.

- LESSCONNECTION(TERMINATE)

All active connectors to the primary structure must also have connectivity to the coupling facility in which the secondary structure is to be allocated. Therefore, any coupling facility that does not provide connectivity for all current active connectors to the structure is dropped from the eligibility list.

When there are no active connectors to the structure, MVS may allow the allocation of the secondary structure in a coupling facility that has less connectivity to systems than does the coupling facility in which the primary structure is allocated. If, at a later time, a connector attempts to connect to the now-duplexed structure and MVS observes that the connector is running on a system that does not have connectivity to both structure instances, MVS will drop the structure out of duplexing. The structure instance that will be kept is that which is accessible to the connector. Note that after the connector connects, MVS may subsequently reduplex the structure into another coupling facility that does provide full connectivity for the set of active connectors to the structure.

- POPULATECF

The concept of “POPULATECF” is not applicable to a duplexing rebuild, and thus does not affect the allocation of a secondary structure. Rebuild processes of any kind cannot be initiated against a duplexed structure.

- Available Space

In order to create a secondary structure that is an exact copy of the primary structure (exact structure attributes and same total and maximum structure counts for all structure objects), it might be necessary for MVS to allocate the secondary structure with a size that significantly differs from that of the primary structure. This is because of different coupling facility storage allocation mechanisms that may exist at different CFLEVELs. Therefore, MVS will process each coupling facility in the preference list as follows:

- Determine the target size (and minimum required control space) of a structure allocated in this particular coupling facility that has the same structure attributes and total and maximum object counts as the primary structure. Note that the maximum structure size from the CFRM policy is not used as an upper bound to the determined target structure; rather, the maximum structure size is allowed to “float” to whatever size is necessary to accommodate the number of structure objects that exist in the primary structure.
- Compare the determined target size and minimum required control space results to the actual free space and free control space in this coupling facility.
 - If the coupling facility has sufficient free space and free control space to accommodate the allocation of the structure, include this coupling facility in the eligibility list.
 - If not, drop this coupling facility from the eligibility list.

- CFLEVEL

System-managed duplexing rebuild requires a coupling facility of CFLEVEL=11 or higher, so any coupling facility in the preference list which is at a lower CFLEVEL is dropped from the eligibility list.

- Volatility

If any active or failed-persistent connectors to the structure requested non-volatility, the system will give preference in the eligibility list to allocating the structure in a nonvolatile coupling facility, using the normal eligibility list weighting for non-volatility.

- Failure-isolation from Connectors

If any active or failed-persistent connectors to the structure requested non-volatility (and thus implicitly requested failure-isolation), MVS will give preference to allocating the secondary structure in a coupling facility that is standalone, that is, failure-isolated with respect to all active connectors to the structure.

Non-standalone coupling facilities, which do not provide failure-isolation from all active connectors, will be allowed to remain in the eligibility list, but behind those that do provide full failure-isolation, using the normal eligibility list weighting for failure-isolation.

- Failure-isolation from Primary Coupling Facility

MVS will give preference to allocating the secondary structure in a coupling facility that is duplex failure-isolated (that is, in a different processor) from the coupling facility in which the primary structure is allocated. Coupling facilities that do not provide failure-isolation from the primary coupling facility will be allowed to remain in the eligibility list, behind all coupling facilities that do provide this failure-isolation. This failure-isolation from the primary coupling facility is a very high-weighted attribute in determining the eligibility list order.

If the secondary structure is allocated in a coupling facility that is not duplex failure-isolated from the primary, MVS issues a highlighted eventual action warning message, IXC553E, to warn about the lack of CF-to-CF failure-isolation.

- Exclusion List

The system prefers to allocate the secondary structure in a coupling facility that does not contain any allocated structure instances (primary or secondary) for any structure listed in this structure's exclusion list. However, there may be instances when a structure is listed in an exclusion list and has already been duplexed. In that case, the following exclusion list processing occurs:

1. The system chooses to allocate the secondary structure in a coupling facility where neither instance of the structure listed in the exclusion list resides.
2. The system chooses to allocate the secondary structure in a coupling facility where one instance of the structure listed in the exclusion list resides, but there is another instance of the duplexed structure in a coupling facility separate from the coupling facility containing the structure awaiting to be duplexed.
3. The system chooses to allocate the secondary structure in a coupling facility where one instance of the structure listed in the exclusion list resides, and the other instance of the duplexed pair does not reside in a separate coupling facility.

Both options 2 and 3 are not considered to have met the exclusion list requirement and therefore the system will set CONAIGNOREDEXCLUSIONLIST ON.

- Application of Pending Policy

The system will not initiate system-managed duplexing rebuild processing while a CFRM policy change is pending against a structure or when, during the process of allocating a duplexed structure, the system discovers that a policy change is pending. When such a policy change is pending, the installation must take whatever action against the structure is required to cause the pending policy change to take effect. This generally involves rebuilding the structure through either a user-managed or system-managed rebuild process, or causing the structure to be deallocated or reallocated.

- Model-dependent Limit on Number of Connectors

It is necessary that all connectors to the primary (or old) structure can be attached to the secondary (or new) structure with the same Connection ID number. The maximum number of connectors to a structure of a given type is a model-dependent coupling facility attribute, and therefore it is possible that some of the coupling facilities in the preference list have a limit that is too low to accommodate all of the attachments that are present in the old structure at this time. Any coupling facility whose model-dependent limit on the number of connectors is insufficient will be dropped from the eligibility list.

Performance enhancements for lock structures with record data

z/OS V1R4 and APAR OW52437 provide performance improvements for lock structures with record data in a system-managed CF structure duplexing environment. Compatibility for the improvements is available for systems in the sysplex at OS/390 V2R8 and higher through APAR OW52266.

The performance improvements may increase the size of an existing lock structure; perhaps an additional 300K of space in the coupling facility may be required. Currently allocated lock structures with record data must be reallocated through the rebuild process to acquire the additional storage in preparation for system-managed CF structure duplexing.

The following procedure outlines the steps to provide compatibility for the performance improvements.

- Apply OW52266 on all systems in the sysplex.
- Use the CF Structure Sizer (CFSizer) tool to determine the appropriate size for all lock structures except ISGLOCK. (The ISGLOCK structure cannot exploit the new performance enhancements because the structure does not contain record data.)

The CF Structure Sizer (CFSizer) tool is found at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062)

- Update the CFRM policy to specify the new lock structure size requirements.

The following procedure outlines the steps to enable the lock structure performance improvements provided in OW52437.

- Apply OW52437 to systems in the sysplex only after OW52266 is installed and active on all systems in the sysplex.
- Initiate® a rebuild to activate the performance enhancements with multiple record data lists.
- When the rebuild completes, issue the DISPLAY XCF,STR command with at least one structure name to verify that the structure has been allocated with multiple record data lists and is ready for system-managed CF structure duplexing. If so, message IXC360I will contain the insert 'NUMBER OF RECORD DATA LISTS PER CONNECTION: nnn'.

If the message insert does not appear, the structure was allocated by a system that does not have the support provided by OW52437.

For more information, see [System-Managed CF Structure Duplexing \(www.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=ZSW01975USEN\)](http://www.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=ZSW01975USEN).

Common problems and solutions

Table 4 on page 78 provides a reference to identify and solve common problems.

Table 4. Common System-Managed Duplexing Rebuild Problems		
Problem	Possible Causes	Possible Solutions
Format utility (IXCL1DSU) does not recognize SMDUPLEX keyword.	Running on a non-z/OS V1R2 system	Upgrade system to z/OS V1R2
	System-managed duplexing rebuild is disabled	Apply APAR OW41617

Table 4. Common System-Managed Duplexing Rebuild Problems (continued)

Problem	Possible Causes	Possible Solutions
Cannot duplex the structure	CFRM couple data set not migrated correctly	To activate the system-managed duplexing rebuild function, the CFRM couple data set must be migrated. See “CFRM parameters for format utility” on page 319.
	CF-to-CF links not defined	Two coupling facilities of CFLEVEL=11 or higher with CF-to-CF connectivity are required. Issue the D CF command to determine whether the coupling facilities are remotely connected.
	Insufficient coupling facility space	If the coupling facility is space constrained, increase the CF storage. If the structure size is not large enough, increase the structure size.
	Preference list does not contain appropriate coupling facilities	Ensure that the preference list contains at least two coupling facilities of CFLEVEL=11 or higher with CF-to-CF link connectivity.
	Structure is allocated in a downlevel coupling facility	Rebuild the structure to a CFLEVEL=11 or higher coupling facility
	DUPLEX(DISABLED) specified in CFRM policy	Check CFRM policy keyword
	CFRM policy change is pending for the structure	Resolve pending policy change
	Exploiter requirements not met	Ensure exploiter software is at the correct level
	Check messages IXC528I, IXC538I, IXC367I, or IXC574I to determine the actual cause of the problem.	
Structure allocation fails following a CFLEVEL upgrade or CFRM policy change.	<p>The CFRM policy SIZE value is substantially larger than the INITSIZE value.</p> <p>To accommodate the maximum size to which the structure can grow (CFRM policy SIZE value) the CF might have to allocate so much control storage that a structure allocated at the size specified by the policy INITSIZE value has no room for anything else. In particular, the structure might contain an insufficient number of entries and elements for the owning application to use. When SIZE is much larger than INITSIZE, the CF might be unable to allocate a structure with a size of INITSIZE at all.</p>	<p>Increase the INITSIZE (or SIZE, if INITSIZE is not specified) in the CFRM policy until allocation is successful. IBM recommends that the SIZE value be in a range of 1.5 - 2.0 times the INITSIZE value.</p> <p>Use the CF Structure Sizer (CFSizer) tool or the Sizer utility to determine the appropriate structure size for the current CFLEVEL. These tools are available at Coupling Facility sizer (www.ibm.com/support/docview.wss?uid=isg3T1027062).</p>

Table 4. Common System-Managed Duplexing Rebuild Problems (continued)

Problem	Possible Causes	Possible Solutions
The original (old) structure is not allocated in the “most preferred” coupling facility (according to the preference list).	When a structure is a candidate for system-managed duplexing rebuild, the structure allocation algorithm will give preference to coupling facilities at CFLEVEL=11 or higher that have CF-to-CF link connectivity to at least one other coupling facility at CFLEVEL=11 or higher that is also in the preference list and that currently has sufficient space to allocate a secondary instance of the structure. As a result, it is possible that the structure is not allocated in the most preferred coupling facility according to the preference list.	For a structure that exploits system-managed duplexing rebuild, consider placing two coupling facilities at CFLEVEL=11 or higher that have CF-to-CF link connectivity to each other at the beginning of the preference list.
The secondary (new) structure is not allocated in the “second most preferred” coupling facility (according to the preference list).	The secondary (new) structure must be allocated in a coupling facility at CFLEVEL=11 or higher that has sufficient space and CF-to-CF connectivity to the coupling facility containing the primary (old) structure. If the second most preferred coupling facility does not meet this criteria, another coupling facility in the preference list may be used.	Consider placing two coupling facilities at CFLEVEL=11 or higher that have CF-to-CF link connectivity to each other at the beginning of the preference list.
After stopping a system-managed duplexing rebuild, the primary (old) structure is no longer in the “most preferred” coupling facility.	When a system-managed duplexing rebuild is stopped with a request to keep the new structure, the new structure becomes the primary structure. As a result, the primary (old) structure may no longer reside in the most preferred coupling facility. If the structure is re-duplexed, the primary (old) will remain where it is and the new structure may wind up in the most preferred coupling facility. If system-managed duplexing rebuilds are started and stopped multiple times, the location of the structures will flip-flop. This is normal.	No action required.

Encrypting coupling facility structure data

On systems at the z/OS V2R3 level and above, list and cache structure entry and entry adjunct data can be encrypted while the data is being transferred to and from the coupling facility and while the data resides in the coupling facility structure. List and cache structure services use secure cryptographic key tokens obtained from the z/OS Integrated Cryptographic Service Facility (ICSF) software element to encrypt and decrypt user provided structure data. Encryption of structure data can be controlled on a structure-by-structure basis.

Steps for setting up and using encrypted coupling facility structures

The following topics describe the system requirements, set up steps and considerations to evaluate when enabling coupling facility structure data encryption on systems and the sysplex:

- [“System requirements” on page 81](#)
- [“XCF requirements” on page 82](#)

- [“Sysplex coexistence considerations” on page 82](#)
- [“Systems joining a sysplex containing encrypted structures” on page 82](#)
- [“Dumping encrypted CF structures” on page 82](#)
- [“Master key changes” on page 83](#)
- [“Structure encryption key management” on page 83](#)
- [“Determining the encryption state of a structure” on page 83](#)

System requirements

- A system must be at z/OS V2R3 or higher operating on IBM zEnterprise EC12 or IBM zEnterprise BC12 or newer servers with the cryptographic hardware configured and activated to perform cryptographic functions and hold Advanced Encryption Standard (AES) master keys within a secure boundary. Cryptographic hardware features available on the required servers that support cryptographic functions and AES master keys include:
 - IBM zBC12 and zEC12 with the Crypto Express3 Coprocessor (CEX3C) or Crypto Express4 Coprocessor (CEX4C)
 - IBM z13 and z13s with the Crypto Express5 Coprocessor (CEX5C).

Note: Feature 3863, CPACF DES/TDES Enablement must be installed to use features CEX3C, CEX4C or CEX5C.

- ICSF is required to be started on the z/OS system.

CP Assist for Cryptographic Functions (CPACF) in the z/OS environment is used to encrypt and decrypt structure data. ICSF works with the hardware cryptographic feature to provide secure, high-speed cryptographic services in the z/OS environment. ICSF is required to be started on the z/OS system to:

- Provide the required ICSF APIs needed by XCF to obtain and manage secure cryptographic tokens for structures, and
- To set and manage AES master keys in the hardware cryptographic feature. AES master keys must be set on all servers where z/OS systems use CF encryption. The AES master key must be the same for all z/OS systems in the sysplex.

XCF requires the use of the ICSF coordinated change master key function and the use of a single sysplex-wide shared cryptographic key data set (CKDS) by all members in the sysplex to ensure CFRM policy data for encrypted structures remain consistent with AES master keys in the sysplex.

For information on setting and managing AES master keys and running ICSF in a sysplex environment, see the sections on CKDS management in a sysplex and coordinated change master key and coordinated refresh in [z/OS Cryptographic Services ICSF Administrator's Guide](#).

For information on installation, initialization, and customization of ICSF and activating coprocessor features, see [z/OS Cryptographic Services ICSF System Programmer's Guide](#).

- The following Access Control Points (ACP) must be enabled in the domain role for the cryptographic coprocessor. The required ACP are enabled by default:
 - Key Generate – OP (Enabled by default)
 - Key Test (Always enabled, can not be disabled)
 - High-performance secure AES keys (Enabled by default)

For information on displaying and setting the Common Cryptographic Architecture (CCA) coprocessor domain role, see [z/OS Cryptographic Services ICSF Administrator's Guide](#), section [Display CCA domain roles](#) and [z/OS Cryptographic Services ICSF Application Programmer's Guide](#), section [Access control points and callable services](#).

XCF requirements

- The XCF address space must have adequate authorization through the z/OS Security Server (RACF) or an equivalent security product to access ICSF services protected by the CSFSERV general resource class. See the information on assigning the RACF TRUSTED attribute in [z/OS MVS Initialization and Tuning Reference](#) for information about using RACF to assign the TRUSTED attribute to the XCF address space.
- The Administrative Data Utility (IXCMIAPU) must be used to create or update structure definitions in the CFRM policy data. The ENCRYPT parameter with the value of YES (e.g., ENCRYPT(YES)) must be added to the structure definition statement of each structure that encryption is desired for.

Note the following:

- In addition to authorizing use of the utility (see [“Authorizing use of the utility”](#) on page 335) for updating CFRM policies, and CFRM couple data sets, at least READ access to ICSF CSFSERV CSFKGN and CSFSERV CSFKYT resource profiles is required when running the utility for the purposes of specifying ENCRYPT(YES) on a structure definition. At least READ access to the CSFSERV profiles is required regardless of whether the utility is run against the ACTIVE CFRM couple data set or an INACTIVE CFRM couple data set. For information on the CSFSERV general class resource, see the [z/OS Cryptographic Services ICSF Administrator's Guide](#).
- The Administrative Data Utility must be run on a system where ICSF is started.

The AES master key for the system where the utility is run must be the same AES master key as the target sysplex where the CFRM administrative data set will be active.

See Chapter 12, [“Administrative data utility,”](#) on page 335 and [“CFRM parameters for administrative data utility”](#) on page 346 for information on the ENCRYPT parameter, adding and updating CFRM policy structure definitions and activating a CFRM policy.

Sysplex coexistence considerations

In a sysplex where encrypted structures are used, but not all systems support connecting to encrypted structures (systems at z/OS V2R2 or below), coexistence support must be installed on the down level systems. Coexistence **APAR OA52060** allows lower level systems to use CFRM couple data sets containing cryptographic information without inadvertently corrupting the cryptographic information set by up level encrypted structure support.

Systems joining a sysplex containing encrypted structures

Ensuring all systems in the sysplex satisfy the system requirements for coupling facility structure encryption also applies to systems joining the sysplex. Special planning considerations need to be made to ensure a system can join a sysplex that has encrypted structures allocated. Specifically, planning is needed to ensure that the AES master key can be set correctly in the hardware cryptographic feature. The AES master key may not have been set correctly for a new system or for a system that was not active when the AES master key was changed. When coupling facility structure encryption is used for critical sysplex functions such as XCF signaling, the system may not be able to be brought up or even join the sysplex without the correct AES master key set. In order to use the z/OS system to set the AES master key, the system may need to run in XCF-local mode first. Planning is needed to ensure a proper configuration to run in XCF-local mode is available. See [z/OS Cryptographic Services ICSF Administrator's Guide](#) for alternatives to using XCF-local mode to set the AES master key.

Dumping encrypted CF structures

The system will decrypt encrypted coupling facility structure data being written to a dump data set so that coupling facility structure dump data will always be unencrypted. STRLIST dump support will recognize that the structure contains encrypted data and decrypt encrypted structure data *before* storing the data in the SVC dump data sets. To ensure data is not stored in the clear in a dump data set, plan to encrypt dump data sets. When planning to encrypt dump data sets, be sure to consider that structure dumps can be taken early in IPL.

Master key changes

Cryptographic key tokens are stored in the CFRM couple data set (CDS). The cryptographic key tokens contain encryption keys enciphered under the current master key. This enciphered encryption key is referred to as a secure key. ICSF will issue a message and issue an ENF (ENF 19) when master key change processing completes. Triggered by that ENF, CFRM will re-encipher all secure keys in the active and the administrative policy records in the active CFRM CDS using the new master key. Message IXC121I is issued when this processing completes.

The use of the ICSF coordinated change master key function and the use of a single sysplex-wide shared CKDS by all members in the sysplex ensures CFRM policy data for encrypted structures remain consistent with AES master keys in the sysplex.

With CFLEVEL22 or above, the structure's secure key is stored in the CF. After the CFRM CDS is updated with the re-enciphered secure keys, CFRM will update the secure keys in the CF. Message IXC121I is issued when this processing completes.

Any inactive/spare CFRM couple data sets will not automatically be updated when a master key change takes place. To re-encipher the secure keys in an inactive CFRM CDS, run the administrative data utility (see “Using the administrative data utility” on page 335) to replace or add a policy that uses ENCRYPT(YES) on a structure definition statement. The utility will issue message IXC747I to indicate that the structure secure keys were re-enciphered under the current master key.

Structure encryption key management

Each structure definition in a CFRM CDS that specifies ENCRYPT(YES) requires an encryption key. The administrative data utility for CFRM must be run to assign an initial encryption key to a structure definition.

The SETXCF MODIFY command can be used to change a structure's encryption key in the active CFRM policy. See the SETXCF MODIFY command in *z/OS MVS System Commands* for more information on how to perform structure key token changes. A new encryption key will be generated for the structure, encrypted under the current master key and stored in the active CFRM CDS. The structure key change will take effect immediately for an unallocated structure. The change will become a pending CFRM policy change for an allocated structure. Additionally, the newly generated structure key will be propagated to all the defined administrative policies in the active CFRM CDS where the structure name exists.

You can run the administrative data utility for CFRM against an inactive/spare CFRM CDS to generate a new encryption key for a structure name by deleting and defining ALL policies in the CFRM CDS where the structure name definition exists. This technique is useful when planning to use an inactive/spare CFRM CDS on a subsequent sysplex-wide IPL or SETXCF COUPLE, PCOUPLE command.

Determining the encryption state of a structure

If not all system and XCF requirements are met, structure data may not be able to be encrypted for an allocated structure. For example, the cryptographic coprocessor configuration may be in error thus preventing CFRM from obtaining the necessary secure key token verification patterns required to perform data encryption.

- The DISPLAY XCF, STR command can be used on z/OS V2R1 and z/OS V2R2 systems with APAR OA52060 and above to identify the coupling facility structure data encryption state of allocated structures and whether the resident structure data state matches the CFRM policy specification. For more information on the DISPLAY XCF,STR command and the ENCRYPTED, NOTENCRYPTED and ENCMISMATCH filter options, see *z/OS MVS System Commands*.
- Beginning in z/OS V2R3, IBM Health Checker for z/OS check XCF_CF_STR_ENCRYPT verifies that structure data in an allocated structure is consistent with the effective ENCRYPT parameter and cryptographic encryption key for the structure (when the structure data is encrypted) in the CFRM policy.

The health check can also be run in verbose mode to list the structure data encryption specification from the CFRM policy and the resident data format (encrypted or not encrypted) for allocated

structures. For more information on check XCF_CF_STR_ENCRYPT, see *IBM Health Checker for z/OS User's Guide*.

The role of CFRM in a sysplex

Coupling facility resource management provides the services to manage coupling facility resources in a sysplex. This management includes the enforcement of CFRM policies to ensure that the coupling facility and structure requirements as defined in each policy are satisfied.

A Note about Operations

Throughout its management of coupling facility resources, CFRM issues operator messages to record the status of its processing. This topic is included so that you will understand the messages that are issued and be prepared to respond if necessary. Your operations staff should have a thorough understanding of the connections to the coupling facility and of the application's use of one or more structures in the coupling facility.

CFRM uses the CFRM couple data set to maintain status data about the use of coupling facility resources in the sysplex and to hold its copy of the active CFRM policy. Whenever a CFRM couple data set is made available in the sysplex, either for the first time or when a switch is made to a different CFRM couple data set, CFRM becomes operational. Once operational, CFRM performs the following services for the sysplex:

- Cleans up status data
- Performs system level initialization
- Reconciles active policy data with data in the coupling facility.

Cleaning up status data

Policy data is maintained on the CFRM couple data set, and therefore remains on the couple data set from one IPL to the next or from one use of the CFRM couple data set to the next. As part of its initialization, CFRM cleans up any status data in the CFRM couple data set that remains in what had previously been the active policy. The cleanup includes:

- Indicating that no system has access to any coupling facility.
- Setting the connections to coupling facility structures to a failed-persistent state (thus preventing connections from accessing those structures).

Performing system level initialization

CFRM performs system level initialization based on the state of the active CFRM policy. Valid CFRM policy states are:

- The CFRM couple data set contains an active policy that is empty, that is, there is no currently active CFRM policy, either a policy that was never started or policy usage was stopped. A policy is activated either by issuing a SETXCF START,POLICY command or specifying CFRMPOL(POLICY-NAME) in the COUPLExx parmlib member used when the sysplex is IPLed. Policy usage is stopped by issuing a SETXCF STOP,POLICY command.
- The CFRM couple data set contains an active policy that was properly defined with the administrative data utility and then activated either by using the SETXCF START command or the CFRMPOL parameter in the COUPLExx parmlib member.

If there is no currently active CFRM policy, any application requests to use coupling facility resources will be denied. It might be necessary for the system programmer to define new administrative policies with the IXCMIAPU utility and then activate a CFRM policy.

If there is a currently active CFRM policy, each system in the sysplex that has access to the CFRM couple data set determines whether it has connectivity to each of the coupling facilities listed in the active policy and records that information. During this phase of CFRM system level initialization when a coupling facility is first used by a sysplex, CFRM gains ownership of the coupling facility for the sysplex.

Ownership is established by setting the coupling facility authority data with ownership information (date, time, and sysplex name) and saving the ownership information in the CFRM active policy. This ownership information is used to determine the need to prompt the operator before a sysplex assumes ownership of a coupling facility. A coupling facility can be used by only one sysplex at a time, thus providing for data integrity and preventing unintentional data loss.

The rules for operator prompting are:

1. Coupling facility is not owned because the coupling facility has zero for authority data. Here the coupling facility cannot be in use by any sysplex so CFRM proceeds to gain ownership of the coupling facility by setting nonzero authority data in the coupling facility. The nonzero value is the sysplex name and a time stamp to reflect when ownership was established. This value is saved in the CFRM active policy along with system connectivity status. No operator prompting.
2. Coupling facility is owned (nonzero authority data) and the value matches the value in the CFRM active policy. Here it is assumed that the coupling facility was last used by the same sysplex so CFRM takes ownership again. No operator prompting.

Note: Specifying or defaulting to NO for CFRMOWNEDCFPROMPT causes the ownership information in the CFRM active policy to remain unchanged when the CFRM couple data set is first used by the sysplex. The saved ownership information can match the nonzero coupling facility authority data.

3. Coupling facility is owned (nonzero authority data) and the value does not match the value in the CFRM active policy. Here CFRM prompts the operator and only gains ownership if the operator response to the messages indicate that the coupling facility can be used by the sysplex.

Note: Specifying YES for CFRMOWNEDCFPROMPT causes the ownership information in the CFRM active policy to be cleared when the CFRM couple data set is first used by the sysplex. The cleared ownership information will not match the nonzero coupling facility authority data.

APAR OA10197 added the CFRMOWNEDCFPROMPT parameter to the COUPLE statement specified in the COUPLExx parmlib member.

Once a sysplex has established ownership of a coupling facility and the coupling facility remains defined in the CFRM active policy, the sysplex will reestablish ownership without operator prompting when a system gains connectivity after a total loss of connectivity to the coupling facility from all systems in the sysplex.

Reconciling CFRM policy data

Once a sysplex has gained ownership of a coupling facility, CFRM continues its initialization by reconciling the data being maintained about the current coupling facility environment and, if necessary, by cleaning up the dump space in the coupling facility.

• Reconciling the data

The system maintains data about use of coupling facility resources in both the active policy and in the coupling facility itself. The coupling facility data is assumed to be more accurate. The system performs the reconciliation process for each coupling facility for which the sysplex has gained ownership. The process consists of comparing data about use of the coupling facility resources from the coupling facility with the data in the active policy. The reconciliation actions taken by CFRM might include deallocating structures that are not persistent and have no persistent connections. CFRM also checks that all persistent structures are listed in the active policy, and must determine how to handle structures or connections remaining in the coupling facility but not in the active policy. The operator might be required to request the deletion of these structures, or CFRM might reconstruct the active policy by adding the structures or connections.

Reconciliation processing ensures that structure definitions and coupling facility structures indicating encrypted structure data are consistent. If discrepancies are found between the CFRM policy and the coupling facility data regarding the expected and actual encryption state of structure data, a structure will be deallocated. To further assist in reconciling of encrypted structures, structures allocated in a coupling facility at CFLEVEL 22 or higher contain additional cryptographic information stored by the system that lower level coupling facilities do not contain.

- **Cleaning up the dump space**

If there were dumps in progress at the time of transition to a new active policy, the system dumps all captured data to a dump data set.

If the operator had issued a CHNGDUMP,STRLIST command to initiate one or more new structure dumps, the dump contains information for each structure listed on the STRLIST parameter.

Transitioning to a new CFRM policy

SETXCF START,POLICY is the command used to activate an administrative policy. If no current active policy is available, the activation takes effect immediately. If a transition to the newly activated policy is required, this command causes the following to occur over time in the active policy.

- When adding a new coupling facility, the logical name of the coupling facility is updated in all the preference lists of the structure definitions in the active policy. There might be a delay while all systems complete the update before an operator can use the new logical name. Once all systems recognize the new coupling facility logical name, the operator can use the name on all commands referencing the coupling facility.
- When changing the dump space size, the size of the coupling facility dump space is updated in the CFRM active policy immediately. However, the actual size of the dump space at any given time is determined by:
 - The size specified in the policy
 - The amount of space assigned to a dump in the dump space
 - The total amount of space available in the coupling facility.

MVS attempts to satisfy requests to change the dump space size immediately and if not successful, continues to attempt the change. Use the DISPLAY CF command to determine the policy size and the actual size of the dump space in the coupling facility.

- When deleting or modifying a structure or coupling facility definition, the policy transition proceeds as follows. Note that the system handles a modification to a structure or coupling facility definition as an implicit “delete and add”.
 - The change takes effect immediately if coupling facility resources are not allocated for the particular structure.
 - Changes to some parameters (such as REBUILDPERCENT, FULLTHRESHOLD, or ALLOWAUTOALT) take effect immediately.
 - The change remains pending if coupling facility resources are allocated either for the structure or for any structures in the coupling facility being modified. The structure's resources need to be deallocated in order for the pending policy change to take effect. Rebuilding the structure is the preferred method of accomplishing this, if the structure allows rebuild. The operator command SETXCF START,REBUILD can be used to rebuild the structure in either the same coupling facility or in another coupling facility. In some cases, it might be necessary to deallocate the structure using the SETXCF FORCE command, but that is not the preferred method.

The DISPLAY XCF command provides specific information about a structure and any pending policy changes.

- When releasing coupling facility resources, the following order should be observed:

1. Remove all connections to an allocated structure.

The connection removal should be initiated by the application using the connection and should result in no active connections to the structure. However, operator intervention might be required to remove failed-persistent connections. Use the DISPLAY XCF,STR command to determine which connections are in a failed-persistent state and then delete each connection using the SETXCF FORCE command.

2. Remove the allocated structure from the coupling facility.

The system deletes a non-persistent structure once all connections to the structure are removed. The operator can remove a persistent structure by using the SETXCF FORCE,STRUCTURE command. Note that if the structure has an associated structure dump, then the operator must issue SETXCF FORCE,STRDUMP before deleting the structure.

- The addition of a coupling facility or structure takes effect immediately, assuming that all resources from the prior active policy have been released. The CFRM couple data set is formatted to contain a fixed number of coupling facility and structure definitions in the active policy. If deletions or modifications of prior coupling facilities or deletions of structures are still pending, then additions might also become pending.

Resource allocation in the coupling facility

Coupling facility structure size is defined in terms of the total storage in the coupling facility to be used. This includes both control areas required by the coupling facility control code and data areas used by the application. Determination of the size is further affected by coupling facility allocation rules and the coupling facility allocation increment size, a function of the level of coupling facility control code.

You must take all of these factors into account when determining how to define your CFRM policy and how to configure your coupling facility.

Coupling facility storage considerations

The storage for a coupling facility LPAR is defined in the same way as a non-coupling facility partition. However, the storage in a coupling facility LPAR cannot be dynamically reconfigured. If another partition on the same processor fails, its storage cannot be taken over by the coupling facility partition. Or, if the coupling facility partition fails, its storage cannot be taken over by another partition.

You can also configure storage-class memory to a coupling facility LPAR. As with coupling facility real storage, you cannot dynamically reconfigure storage-class memory.

The DISPLAY CF command displays information about coupling facility storage and storage-class memory, including the total amount, total in-use, and total free storage. The RMF Coupling Facility Usage Summary Report provides a more permanent copy of this information.

Coupling facility resource allocation “rules”

A coupling facility structure is located in a particular coupling facility and allocated at a certain size depending on:

- Values specified in the CFRM policy specification
- Minimum size as determined by the coupling facility
- Authorized application specification when using the XES services
- Coupling facility storage increment.

CFRM policy specification

The CFRM policy contains the maximum structure size and can contain a smaller initial size also. The initial structure size defined in the CFRM policy (or the maximum size if an initial size is not specified) is used as the attempted allocation size unless it is overridden by a structure size specified on the IXLCONN macro.

The CFRM policy can also optionally designate a minimum structure size. The MINSIZE value specifies the minimum bound for the structure size on all allocation requests except those resulting from system-managed rebuild.

The CFRM policy can optionally specify the maximum amount of storage-class memory that is available to a structure. The use of storage-class memory affects the structure's resource usage in a complex manner. See [“Requesting the use of storage-class memory” on page 54](#) for a discussion of its implications.

Authorized application specification

When requesting an XES service to connect to a structure, the application optionally can specify a size for the structure. The system uses the smaller of the two sizes (as specified in the CFRM policy or by the application), as the target allocation size for the structure.

The application also is required to specify certain structure attributes when connecting to a structure. These structure attributes are used by the coupling facility control code when determining how to most efficiently allocate the various parts of the structure in the coupling facility. Some examples of structure attributes are data element size, whether locks are used, the number of list headers, and whether an adjunct area is required. The coupling facility control code evaluates each attribute in the following sequence:

- **Available space**

The structure is allocated as large as possible based on the available storage in the requested coupling facility. The target size is derived from either the CFRM policy or the application's request to connect to the structure.

- **Fixed structure controls**

Storage for the fixed structure controls used by the coupling facility is allotted first from the total available structure storage. The amount of storage required for the structure controls is dependent on the CFLEVEL of the coupling facility and the maximum structure size defined in the CFRM policy. Therefore, it is important to consider both dependencies when specifying structure size. If too large a portion of the available storage is allotted to the structure controls, there may not be enough storage left for allocation of the remaining structure objects, and the structure in effect will be unallocatable or unusable.

- **Entry/element ratio**

Within the remaining structure storage, once the fixed structure controls have been allocated, the coupling facility control code attempts to allocate entries and elements in a way that most accurately approximates the requested entry/element ratio. If the structure has been allocated with a size less than the minimum structure size required by the specified structure attributes, then the entry/element ratio may deviate from the requested value. If the structure has been allocated with a size greater than or equal to this minimum structure size, then the entry/element ratio should be satisfied.

- **Entry and element counts**

Within the remaining structure storage, once the fixed structure controls have been allocated, the coupling facility control code attempts to maximize the actual number of entries and elements (as specified by the ratio) that can be placed in the structure.

For a description of the structure attributes that an authorized application can specify, see [z/OS MVS Programming: Sysplex Services Guide](#) and [z/OS MVS Programming: Sysplex Services Reference](#).

Coupling facility storage increment

Coupling facility storage is allocated in multiples of the coupling facility operational level (CFLEVEL) storage increment size. See *PR/SM Planning Guide* for a list of the coupling facility limits for CFLEVEL.

Understanding coupling facility use of processor resources

Both the cache and the list structure use resources in the processor to which the coupling facility is attached. These resources, in HSA, are allocated to the processor at processor initialization time and cannot be modified during sysplex processing. Specifically, the structure use of processor resources is:

- The local cache vector is a user-defined vector that provides a way for cache structure users to determine the validity of data in their local cache vectors. There is one local cache vector per user of the cache structure. Each vector is divided into separate entries with each entry corresponding to a local cache buffer. Each vector entry contains an indicator that the system sets to indicate whether the data in the corresponding local cache vector is valid. Users must test the indicator to determine the validity of the data in their local cache buffers.

- The list notification vector is a user-defined vector that provides a way for list structure users to determine the state (empty or nonempty) of each list or event queue they are monitoring. Each connector to a list structure that indicates interest in list or event queue monitoring is allocated a list notification vector. The vector consists of an array of entries, each of which can be associated with a particular list header or the user's event queue. Each entry contains an indicator that the system sets to indicate the empty or nonempty state of the list or event queue.

XES provides the IXLVECTR service to examine the contents of the vectors allocated in HSA, but the service does not allow you to free up the vectors. Vectors that are allocated in HSA are freed up when connectors disconnect, but remain there in error scenarios such as when a system failure occurs. The implication of having obsolete vectors remaining in HSA is that the subsystems that require the allocation of vectors might not be able to initialize if additional HSA storage is not available.

To clear the obsolete vectors from HSA, the system must be reset with either a LOAD or SYSRESET command. On all processors, the LOAD CLEAR or SYSRESET CLEAR command can be used to clear the unused, obsolete vectors from HSA. On some processors, the LOAD NORMAL and SYSRESET NORMAL function has been enhanced to clear the vectors from HSA.

- If you are running on a processor that has not been upgraded to allow the clearing of obsolete vectors from HSA, use the LOAD CLEAR or SYSRESET CLEAR command to clear the vector space.
- If you are running on a processor that has been upgraded to allow the clearing of obsolete vectors from HSA, use the LOAD NORMAL or SYSRESET NORMAL command to clear the vector space.

See the applicable Operations Guide for your processor for a description of the LOAD and SYSRESET commands.

Operating in a coupling facility environment

The coupling facility is designed to be run with little operator intervention. There are times, however, such as system-level initialization, when the operator is required to make decisions about how coupling facilities or structures are to be handled in the sysplex. A point to remember is that the coupling facility might contain important MVS or application data or structures, so precautions must always be taken to preserve that data.

Some instances when operator intervention might be required for a coupling facility in a sysplex are:

- When a coupling facility needs to be removed from a sysplex, for maintenance for example.
- When a coupling facility needs to be transferred to another sysplex.
- When a coupling facility needs to be repopulated with the most appropriate structures.

Chapter 17, “Coupling Facility Guidelines,” on page 425 provides specific details and examples for operating in a coupling facility environment.

Removing a coupling facility from a sysplex

A coupling facility might need to be removed from a sysplex for maintenance, to have a new level of coupling facility control code installed, or to be replaced. When such a situation occurs, you must remove any structures contained in the coupling facility before it is shut down and removed from the sysplex.

See “Sample procedure for coupling facility maintenance” on page 92 for steps to take in preparation for removing a coupling facility from a sysplex, such as putting the coupling facility in maintenance mode, and moving all the structures out of the coupling facility. By placing a coupling facility into maintenance mode, the XCF allocation algorithm will eliminate it from the list of coupling facilities that are eligible for structure allocation. Once the coupling facility is in maintenance mode, the REALLOCATE process can be used to remove structures from it.

See Chapter 17, “Coupling Facility Guidelines,” on page 425 for procedures for shutting down a coupling facility.

Transferring a coupling facility to another sysplex

You might need to transfer a coupling facility from one sysplex (sysplex A) to another (sysplex B) in order to do maintenance on sysplex A. The procedure for doing the coupling facility transfer includes:

- Move all structures out of the coupling facility owned by sysplex A.
- Remove the coupling facility from sysplex A's CFRM policy and all preference lists in sysplex A.
- Add the coupling facility to sysplex B's CFRM policy and one or more preference lists in sysplex B.

This procedure allows the system to grant ownership of the coupling facility to sysplex B without requiring further operator intervention after the initial structure removal.

Using REALLOCATE or POPULATECF to relocate coupling facility structures

Note:

1. The REALLOCATE process and the POPULATECF function are mutually exclusive.
2. When a coupling facility is not eligible for structure allocation, it cannot be the target of a POPULATECF function and the REALLOCATE process will tend to relocate structures to other coupling facilities. The XCF allocation algorithm recognizes that a coupling facility is not eligible for structure allocation when it is:
 - In maintenance mode
 - In cleanup
 - Is pending deletion from the active CFRM policy
 - Has failed

POPULATECF function

The POPULATECF function provides a means of rebuilding multiple structures into a coupling facility with a single operator command or IXLREBLD macro invocation. The function's use primarily is to repopulate a coupling facility with the most appropriate structures after the coupling facility has been added to the sysplex configuration or has been removed from the configuration for service and then returned. The advantage of the POPULATECF function is that it rebuilds only structures that are currently allocated into the coupling facility that the structure (in the CFRM active policy) has specified is preferable. If the structure already resides in a coupling facility that is higher in its preference list than the coupling facility being populated, the system will not rebuild the structure.

For example, assume that a coupling facility structure is to be moved from a standalone coupling facility to a machine containing only a coupling facility LPAR. If the machine has the capability of also running z/OS in another LPAR, the system would not rebuild the structure in that machine. The standalone coupling facility configuration is considered "more suitable" and therefore higher in the preference list than the non-standalone coupling facility configuration.

"Sample procedure for coupling facility maintenance" on page 92 outlines the steps an operator would take when having to remove a coupling facility from the configuration for maintenance and then subsequently returning it to service. In the example, coupling facility CF1 is to be removed, leaving coupling facilities CF2 and CF3 in the configuration. The structures in CF1 must be moved to one of the other coupling facilities before CF1 can be removed from the configuration.

REALLOCATE process

The REALLOCATE process provides a means to evaluate each allocated structure and to adjust as needed the location of structure instance(s) with a single operator command. Like POPULATECF, the primary usage is to populate a coupling facility that has been brought into use in a sysplex. Unlike POPULATECF, the REALLOCATE process can adjust the location of both simplex and duplexed structures. The advantage of the REALLOCATE process is that it uses structure allocation criteria to determine whether to initiate structure rebuild processing to move allocated instance(s) into coupling facilities listed in the preference

list from either the active or pending CFM policy. If the structure instance(s) already reside in the "more suitable" coupling facility, the system will not initiate structure rebuild processing.

The XCF allocation algorithm builds a list of eligible CFs using the structure's preference list and the state of the CF. Coupling facilities that are in maintenance mode, pending delete from the CFM active policy, in cleanup, or failed are eliminated from the eligibility list. When a CF is in any of these states, it is considered less suitable. You can place a coupling facility in and out of maintenance mode using the SETXCF START MAINTMODE and SETXCF STOP MAINTMODE commands. See ["Coupling facility resource allocation "rules""](#) on page 87 and ["Sample procedure for coupling facility maintenance"](#) on page 92.

For example, assume that a coupling facility structure is duplexed with the old instance in the third CF listed in the preference list and the new instance in the first CF listed in the preference list. If evaluation using the structure allocation criteria determines that the "more suitable" location for the structure instances should have the old instance in the first CF listed and the new instance in the second CF listed, then the REALLOCATE process would take two steps to adjust the location by stopping duplexing keeping the new instance followed by reduplexing the structure. The result would have the old instance in the first CF listed in the preference list and the new instance in the second CF listed.

You can use the SETXCF START,REALLOCATE operator command to do the following:

- Move structures out of a CF following a CFM policy change that deletes/changes that CF (for example, in preparation for a CF upgrade).
- Move structures out of a CF following placing the CF into maintenance mode.
- Move structures back into a CF following a CFM policy change that adds/restores the CF (for example, following a CF upgrade/add).
- Move structures back into a CF following taking a CF out of maintenance mode.
- Clean up pending CFM policy changes that may have accumulated for whatever reason, even in the absence of any need for structure "relocation" per se.
- Clean up simplex or duplexed structures that were allocated in or moved into the "wrong" CF(s), for whatever reason (for example, the "right" CF was inaccessible at the time of allocation).
- Clean up duplexed structures that have primary and secondary "reversed" due to a prior condition which resulted in having duplexing stopped with KEEP=NEW and the structure reduplexed.

Note: REALLOCATE processing may require that all systems have connectivity to all coupling facilities.

Before starting a REALLOCATE process, you can use the DISPLAY XCF,REALLOCATE,TEST command to request information (through message IXC347I) about what can be expected from subsequent REALLOCATE processing.

You can use the DISPLAY XCF,REALLOCATE,REPORT command to request detailed information (through message IXC347I) about the most recent REALLOCATE process. If a REALLOCATE process ends with an exception condition, the system automatically issues the DISPLAY XCF,REALLOCATE,REPORT command internally.

Comparison of POPULATECF function and REALLOCATE process

Table 5 on page 91 shows a comparison of the POPULATECF and REALLOCATE functions:

Table 5. Comparison of POPULATECF and REALLOCATE	
POPULATE function	REALLOCATE process
Use SETXCF commands or IXLREBLD macro to start or stop.	Use SETXCF commands to start or stop.
Only supports simplex structures.	Supports both simplex and duplexed structures.
Uses the position of the specified coupling facility in the preference list of a structure to select it as a candidate structure.	Evaluates current location of structure instance(s) based on structure allocation criteria to select it as a target structure.

Table 5. Comparison of POPULATECF and REALLOCATE (continued)	
POPULATE function	REALLOCATE process
No structure-level control to preclude selection as a candidate structure.	ALLOWREALLOCATE(NO) specified on STRUCTURE definition in the CFRM policy provides structure-level control to preclude selections as a target structure.
Predetermines the set of simplex structures which are marked pending rebuild and lists them in message IXC540I. Each candidate structure will be processed serially to completion before the next structure is processed. The serial nature of the processing allows even XCF signalling structures to be selected.	Does not predetermine the set of structures but selects a target structure, takes necessary steps, then proceeds to examine next allocated structure. The serial nature of the processing allows even XCF signalling structures to be selected.
Only a single step (rebuild which can be user-managed or system-managed) is used to relocate the structure.	The number of steps used to adjust the location is based on whether the structure is simplex or duplexed.
Only the specified coupling facility is considered for allocating the new instance.	The relocated instance(s) may be allocated in any in-use coupling facility which is selected based on the structure allocation criteria.
Does not process a structure when the only reason for processing it would be to cause a pending CFRM policy change to take effect.	Processes a structure even when the only reason for processing it is to cause a pending CFRM policy change to take effect (that is, attempts to clear all pending CFRM policy changes for allocated structures).

Sample procedure for coupling facility maintenance

The following examples outline some steps an operator can take to prepare a coupling facility for removal from the configuration and then subsequently return it to the configuration after maintenance procedures have been completed. In the example coupling facility CF1 is the coupling facility being removed, leaving coupling facilities CF2 and CF3 in the configuration. To prepare for this, the structures in CF1 must be moved to one of the other coupling facilities before CF1 can be removed from the configuration.

1. **Place the coupling facility in maintenance mode:** In preparation for removing coupling facility CF1 from the sysplex, you can place it in maintenance mode. This step prevents systems from allocating any structures in CF1 while you prepare to take it down for upgrade procedures. Issue operator command:

```
SETXCF START,MAINTMODE,CFNAME=CF1
```

2. **Move structures out of CF1:** The operator can move structures in CF1 to another coupling facility with any of the following commands:

- SETXCF START,REBUILD,CFNAME=CF1,LOCATION=OTHER

The LOCATION=OTHER specification indicates that each structure is to be rebuilt in a different coupling facility in each structure's preference list. If no other suitable coupling facility exists in the CFRM policy for a particular structure, it might be necessary to update the CFRM policy so that a coupling facility can be found.

- SETXCF START,REBUILD,STRNAME=stname,LOCATION=OTHER

Note that XCF signalling structure must be rebuilt one at a time.

- SETXCF STOP,REBUILD,DUPLEX,CFNAME=CF1

This method works because when a duplexing rebuild is stopped with CFNAME specified, the instance allocated in the specified coupling facility is NOT kept.

Once all structure have been removed from CF1, it can be removed from the configuration for maintenance.

3. **Take the coupling facility out of maintenance mode:** If you put coupling facility CF1 in maintenance mode, you must take it out of maintenance mode before you try to move structures back into it. If you

don't take it out of maintenance mode first, it will still be on the list of undesirable coupling facilities and it will be impossible to move structures into it.

Use the following command to take a coupling facility out of maintenance mode: Issue operator command:

```
SETXCF STOP,MAINTMODE,CFNAME=CF1
```

4. **Move structures back into CF1 when the coupling facility is back up:** After CF1 is returned to the sysplex configuration, you can use either the REALLOCATE process or the POPULATECF function to move structures back into CF1. The REALLOCATE process and the POPULATECF function are mutually exclusive. Use one of the following options:

- SETXCF START,REALLOCATE

Reallocates structures into all coupling facilities, including CF1, as appropriate based on the current CFRM policy and structure allocation criteria. The REALLOCATE process ensures that only those structure instance(s) allocated in a "less suitable" coupling facility have structure rebuild processing initiated to adjust the location or activate a pending policy change.

Thus, the entire set of allocated structures (simplex or duplexed) in the CFRM active policy are eligible to be evaluated for placement in CF1 and any other coupling facility in use by the sysplex, even if they were not part of the original CF1 evacuation. As each allocated structure in the CFRM policy is examined, messages are issued to the operator or written to the hardcopy log to track the processing as follows:

- IXC574I showing the current location of instance(s), the policy information used, and the results of applying the XCF allocation algorithm.
- When a structure does not have REALLOCATE processing attempted, IXC544I is issued.
- When a structure is the target of REALLOCATE processing, messages for structure rebuild processing (IXC52nI and IXC57nI) and the deallocation process (IXC579I) are issued as the necessary step(s) are taken to adjust the location or activate a pending policy change.

When the REALLOCATE processing finishes, messages IXC545I and IXC543I are issued to the operator.

- SETXCF START,REBUILD,POPULATECF=CF1

Repopulates the CF1 coupling facility with structures. The POPULATECF keyword ensures that only those structures that have CF1 as a higher coupling facility in their preference list than where the structure is currently allocated will be rebuilt in CF1.

Thus, the entire set of allocated simplex structures in the CFRM active policy are eligible to be rebuilt in CF1, even if they were not part of the original CF1 evacuation.

As each structure in the CFRM policy is examined, messages to the operator indicate:

- Whether a structure is eligible to be rebuilt.
- Whether a structure is not eligible to be rebuilt because CF1 is not in the structure's preference list.
- Whether the structure is to remain in its current location because, although the structure might be eligible to be rebuilt, CF1 is not higher in the structure's preference list than that in which the structure already resides or there are other reasons why CF1 is not more suitable.
- The successful rebuild of a structure into CF1.

Note that after the population of CF1, if it is necessary to redistribute the structures for a better sysplex balance, the preference lists in the CFRM policy could be updated. Once the updated CFRM policy is activated, the POPULATECF function could again be used for CF1 or any other coupling facility to redistribute the structures.

Summary of MVS operator commands

Table 6 on page 94 provides a summary of MVS operator commands available to assist in the operation of a coupling facility. The complete syntax of each command is in [z/OS MVS System Commands](#).

Table 6. Summary of MVS commands for managing a coupling facility	
MVS Command	Use for Coupling Facility
CONFIG CHP	Use this command to change the physical status of a CHPID for a coupling facility link. The DISPLAY MATRIX command displays the CHPID information for coupling facility CHPIDs.
DUMP	Use this command to dump cache and list structure information, both control information and data. Control information includes directory/registration information, list controls, list entry controls, storage and castout class controls, and optionally adjunct data; data includes list or cache entry data and optionally adjunct data. Lock structures cannot be dumped.
DISPLAY CF	Displays storage and attachment information about coupling facilities attached to the system on which the command is processed. The request can be for one or more named coupling facilities or all coupling facilities. The information displayed includes: <ul style="list-style-type: none"> • Coupling facility unique identifiers • Device, subchannel, and CHPID status • Utilization of dump space. • Coupling facility level • Volatility status of the coupling facility. <p>DISPLAY CF is the command used to show how coupling facility and structure resources are being used in the sysplex, with the purpose of identifying possible adjustments or modifications to a CFRM policy.</p> <p>With z/OS Release 2, a coupling facility at CFLEVEL=11 and higher will provide information about coupling facilities to which it has CF-to-CF connectivity in the display output section titled "REMOTELY CONNECTED COUPLING FACILITIES". For each remotely connected coupling facility for each "subject" coupling facility, the system will display the following information:</p> <ul style="list-style-type: none"> • The node descriptor (ND) • The cfname (Only displayed when the remotely connected coupling facility is connected to the system on which the DISPLAY CF request is being performed; otherwise, N/A is displayed.) <p>See message IXL150I for a description of the information returned from a DISPLAY CF command.</p>
DISPLAY XCF	
DISPLAY XCF,COUPLE	Use to display XCF information that includes status of optional functions of z/OS (for example, the CFLCRMGMT function) and summary information about couple data sets.
DISPLAY XCF,COUPLE,TYPE=CFRM	Displays information about CFRM couple data sets in use in the sysplex. The information includes: <ul style="list-style-type: none"> • Physical attributes of the CFRM couple data set (name, volume serial number, device address, and time the data set was formatted) • Maximum number of systems that the primary CFRM couple data set can support • Names of the systems using the CFRM couple data set.

Table 6. Summary of MVS commands for managing a coupling facility (continued)	
MVS Command	Use for Coupling Facility
DISPLAY XCF,STRUCTURE,...	<p>Displays information about coupling facility structures in a policy. The information displayed may be summary or detailed information, depending on what options are specified on the command. The information includes:</p> <ul style="list-style-type: none"> • The status of each structure in the active CFRM policy • Whether or not the CFRM couple data set needs to be reformatted, and if so, the number of structures and connections to be used in reformatting • In which coupling facility the structure is allocated, with the identifying characteristics of the coupling facility • The structure size, both as specified in the CFRM policy and the actual size allocated (when detailed information is requested) • Connections to the structure, both maximum allowed and current, with specific information about each connector (when detailed information is requested) • Preference and exclusion list information for the structure (when detailed information is requested) • Pending policy information if a policy change is pending for the structure. <p>When a structure is in a rebuild process, the primary structure is referred to as the “old” structure and the secondary structure is referred to as the “new” structure.</p> <p>See messages IXC359I and IXC360I for a description of the information returned from a DISPLAY XCF,STR command.</p>
DISPLAY XCF,CF,...	<p>Displays information about one or more coupling facilities defined in the active CFRM policy. Summary or detailed information may be obtained. The information includes:</p> <ul style="list-style-type: none"> • Whether or not the CFRM couple data set needs to be reformatted, and if so, the number of structures and connections to be used in reformatting • Name and unique identifiers of each coupling facility as specified by the CFRM policy • Size of the dump space, as specified in the policy (when detailed information is requested) • Storage increment size (when detailed information is requested) • Names of the systems attached to each coupling facility (when detailed information is requested) • Names of the structures allocated in each coupling facility (when detailed information is requested). <p>See messages IXC361I and IXC362I for a description of the information returned from a DISPLAY XCF,CF command.</p>
DISPLAY XCF,POLICY,TYPE=CFRM	<p>Displays information about the CFRM policy in use in the sysplex. The information includes:</p> <ul style="list-style-type: none"> • The name of the active CFRM policy • The status of the policy • The date and time the policy was started • The date and time the policy was last updated.
DISPLAY XCF,REALLOCATE...	Displays information about the REALLOCATE process.
SETXCF	
SETXCF COUPLE,TYPE=CFRM,...	Use this command to control use of the CFRM couple data set — to identify the primary CFRM couple data set (PCOUPLE=), the alternate CFRM couple data set (ACOUPLE=), or to switch from using the primary to using the alternate (PSWITCH).

Table 6. Summary of MVS commands for managing a coupling facility (continued)

MVS Command	Use for Coupling Facility
SETXCF START,...	<p>Use this command to initiate processing for coupling facility resources.</p> <ul style="list-style-type: none"> • SETXCF START,POLICY,TYPE=CFRM is used to activate a CFRM administrative policy in the sysplex • SETXCF START,REBUILD,STRNAME or CFNAME is used to initiate the rebuild process for one or more named coupling facility structures or for all structures in a coupling facility. The system does not start rebuild for XCF signaling structures when the command specifies all structures in a named coupling facility. • The PATHIN and PATHOUT options with STRNAME are used to start the use of a structure as an inbound or outbound XCF signaling path. • SETXCF START,MAINTMODE,CFNAME places a coupling facility in maintenance mode before taking a coupling facility down for maintenance. • SETXCF START,REALLOCATE initiates the REALLOCATE process to recognize the need to relocate structure instances.
SETXCF STOP,...	<p>Use this command to halt the processing that is currently being done for coupling facility resources.</p> <ul style="list-style-type: none"> • SETXCF STOP,POLICY,TYPE=CFRM is used to deactivate the current active policy. • SETXCF STOP,REBUILD,STRNAME or CFNAME is used to discontinue the rebuild process for one or more named coupling facility structures or for all structures in a coupling facility. • The PATHIN and PATHOUT options with STRNAME are used to discontinue the use of a structure as an inbound or outbound XCF signaling path. • SETXCF STOP,MAINTMODE,CFNAME takes a coupling facility out of maintenance mode after returning the CF to the sysplex once maintenance is complete. • SETXCF STOP,REALLOCATE stops an in-progress REALLOCATE process.
SETXCF FORCE,...	<p>Use this command to clean up coupling facility resources in a coupling facility. The resource can be either a structure actively in use in the sysplex, a connection to a structure, a dump associated with a structure, all failed-persistent connections to a structure, or dump serialization for a structure.</p>
SETXCF FUNCTIONS,...	<p>Use the SETXCF FUNCTIONS command to enable or disable optional functions of z/OS (for example, the CFLCRMGMT function).</p>
SETXCF MODIFY,...	<p>Use this command to change structure attributes of a coupling facility structure defined for XCF signaling.</p>
VARY PATH	<p>Use this command to change the state of the logical paths that lead to a coupling facility. The DISPLAY CF command displays the CHPIDs associated with a logical path to a coupling facility. Note that a logical path to a coupling facility is designated by a CHPID and a coupling facility name, not a CHPID and device number.</p>
VARY XCF	<p>Use this command to remove a system from the sysplex through the sysplex partitioning process.</p>

Coupling facility structures for IBM products

Each application that uses a coupling facility structure must provide the information necessary to define the structure in a CFRM policy. In z/OS, information about the following structures for IBM products can be found in the associated documentation.

Table 7. Coupling facility structures in z/OS for IBM products

Structure Function	Structure Type	User-Managed Rebuild	User-Managed Duplexing Rebuild	Alter	System-Managed Processes	Where Documented
XCF signaling	List	Yes	No	Yes	No	See Chapter 5, “Planning signaling services in a sysplex,” on page 99.

Table 7. Coupling facility structures in z/OS for IBM products (continued)

Structure Function	Structure Type	User-Managed Rebuild	User-Managed Duplexing Rebuild	Alter	System-Managed Processes	Where Documented
Note: For communications in the sysplex, JES3 uses the XCF signaling structure if it is defined. As a result, you do not need to define JES3 signaling structures in the sysplex.						
XCF Note Pad	List	No	No	Yes	Yes	See Chapter 6, “Planning XCF Note Pad Services in a sysplex,” on page 141.
VTAM® generic resources function	List	Yes	No	Yes	Yes	<i>z/OS Communications Server: SNA Network Implementation Guide</i>
TCP/IP functions - sysplexports and sysplex wide security associations	List	Yes	No	No	Yes	<i>z/OS Communications Server: SNA Network Implementation Guide</i>
JES2 checkpoint data set in a coupling facility	List	No	No	No	Yes	<i>z/OS JES2 Initialization and Tuning Guide</i>
IMS DBCTL DL/I datasharing with IRLM 2.1 <ul style="list-style-type: none"> IRLM locking OSAM buffer invalidate VSAM buffer invalidate 	<ul style="list-style-type: none"> Lock (IRLM) Cache (OSAM) Cache (VSAM) 	<ul style="list-style-type: none"> Yes Yes Yes 	<ul style="list-style-type: none"> No No No 	<ul style="list-style-type: none"> Yes No Yes 	<ul style="list-style-type: none"> Yes PQ45407 No No 	<i>IMS Version 8: Administration Guide: System</i>
IMS DEDB shared VSO data sharing	Cache	No	No	No	Planned	<i>IMS Version 8: Administration Guide: System</i>
IMS shared message queues	List	Yes	No	No	Planned	<i>IMS Version 8: Administration Guide: Transaction Manager</i>
RACF data base in a large sysplex environment	Cache	Yes	No	No	No	<i>z/OS Security Server RACF System Programmer's Guide</i>
Automatic tape switching	List	Yes	No	No	No	See Chapter 18, “Using automatic tape switching (ATS STAR),” on page 467.
DB2 data sharing	<ul style="list-style-type: none"> Cache Lock List 	<ul style="list-style-type: none"> Yes Yes Yes 	<ul style="list-style-type: none"> Yes No No 	<ul style="list-style-type: none"> Yes Yes Yes 	<ul style="list-style-type: none"> No Yes PQ45407 Yes 	<i>DB2 Data Sharing: Planning and Administration</i>
Log stream data sharing	List	Yes	No	Yes	Yes	See Chapter 10, “Planning for system logger applications,” on page 213.
Global resource serialization star complex	Lock	Yes	No	No	No	<i>z/OS MVS Planning: Global Resource Serialization</i>
CICS® temporary storage data sharing	List	No	No	Yes	Yes	<i>CICS System Definition Guide</i>
CICS TS coupling facility data tables	List	No	No	Yes	Planned	<i>CICS System Definition Guide</i>
CICS TS named counter server	List	No	No	No	Planned	<i>CICS System Definition Guide</i>
VSAM record level sharing	<ul style="list-style-type: none"> Cache Lock 	<ul style="list-style-type: none"> Yes Yes 	<ul style="list-style-type: none"> No No 	<ul style="list-style-type: none"> No No 	<ul style="list-style-type: none"> No Yes 	<i>z/OS DFSMSdfp Storage Administration</i>
Batch Pipes	List	Yes	No	Yes	Planned	<i>BatchPipes OS/390 Users Guide and Reference</i>
Enhanced Catalog Sharing	Cache	No	No	No	No	<i>z/OS DFSMS Managing Catalogs</i>

Table 7. Coupling facility structures in z/OS for IBM products (continued)						
Structure Function	Structure Type	User-Managed Rebuild	User-Managed Duplexing Rebuild	Alter	System-Managed Processes	Where Documented
MQ shared queues	List	No	No		Yes	<i>IBM MQ for OS/390 System Management Guide</i>
WLM multisystem enclaves	Cache	Yes	No	Yes (Size only)	Yes	<i>z/OS MVS Planning: Workload Management</i>
Intelligent Resource Director (LPAR CPU Management, Dynamic Channel Path Management)	Cache	No	No	Yes (Size only)	Yes	<i>z/OS MVS Planning: Workload Management</i>
Note: 1. DB2 and IMS documentation is available in the IMS in IBM Documentation (www.ibm.com/docs/en/ims). 2. CICS documentation is available at CICS Transaction Server for z/OS (www.ibm.com/docs/en/cics-ts).						

Chapter 5. Planning signaling services in a sysplex

Signaling is the mechanism through which XCF group members communicate in a sysplex. In a multisystem sysplex, signaling can be achieved through:

- Channel-to channel (CTC) communication connections, which can be:
 - An ESCON or FICON channel operating in CTC mode
 - The 3088 Multisystem Channel Communication Unit
- A coupling facility (using coupling facility list structures)
- A combination of CTC connections and list structures.

Full signaling connectivity is required between all systems in a sysplex; that is, there must be an outbound and an inbound path between each pair of systems in the sysplex.

To avoid a single point of signaling connectivity failure, IBM recommends redundant connections between each pair of systems, through either CTC connections or coupling facility list structures.

Recommended steps for setting up signaling

The following is the general recommended sequence for setting up signaling in a sysplex.

1. For a multisystem sysplex, define XCF signal paths in the COUPLExx parmlib member. IBM strongly suggests that you configure redundant signal paths to eliminate any single points of failure since systems that lose signal connectivity must be removed from the sysplex. In a base sysplex, this redundancy minimally requires a pair of PATHOUT CTC signal paths and a pair of PATHIN CTC signal paths between every pair of systems in the sysplex. For a parallel sysplex, this redundancy is typically satisfied by defining an XCF signal structure in each coupling facility. If the sysplex has but one coupling facility, at least two XCF signal structures should be defined in the CF. Signal structures are typically defined to XCF as both PATHOUT and PATHIN on every system in the sysplex.
2. Ensure that the signaling configuration constrains XCF message buffer consumption to levels that are safe for the system. Message buffers consume fixed real storage, some of which is backed below the bar. Message buffers also consume virtual storage in an XCF data space that is limited to 2 GB. Both storage areas are used for purposes other than XCF message buffers. The system can suffer an outage if it runs short on frames below the bar or virtual pages in the XCF data space. In general, a given system is protected from excessive message buffer consumption if:
 - a. Its inbound buffer space is limited to 800 MB (the sum of the MAXMSG values for its inbound signal paths is less than 800,000).
 - b. Its outbound buffer space for any one target system is limited to 800 MB (the sum of the MAXMSG values for the transport class definitions plus the sum of the MAXMSG values for the outbound signal paths that are connected to the target system is less than 800,000).

Note: Smaller values can be warranted in configurations where multiple systems might become unresponsive simultaneously, or if the subject system is not passing the `XCF_SYSSTATDET_PARTITIONING` health check in *IBM Health Checker for z/OS User's Guide*.

In general, most sysplex configurations easily satisfy these conditions. Installations with many systems in the sysplex, or many XCF signal paths, or large MAXMSG values might need to limit the number of signal paths or use smaller MAXMSG values to constrain XCF buffer usage to safe levels. For example, a 25-system sysplex with 10 signal structures that use an inbound MAXMSG of 4000 might consume as much as 960 MB of fixed storage (24*10*4000 K) for inbound message buffers.

Note: 800 MB is reasonably conservative and likely safe for many. However, a "safe" value is workload-dependent. Some installations might need to be more restrictive while others survive with a larger limit.

3. Ensure that the XCF function switch XTCSIZE is enabled on all systems in the sysplex (the function is enabled by default). If not, you might need to configure XCF Transport Classes to better optimize the XCF signal service. Any system running with the function disabled might need transport class definitions. Regardless of the XTCSIZE setting, every system in the sysplex might need transport class definitions if there is any one system in the sysplex that does not support the XTCSIZE function.
4. Adjust the number of signal paths and permitted message buffer space as needed to accommodate the workload. For many installations, the signal capacity and throughput needs of the sysplex will be satisfied simply by configuring the signal paths to avoid single points of failure. In general, an XCF signal structure has an expected throughput rate of thousands of signals per second. When the average signal rate between pairs of systems gets to be thousands of signals per second, more care is required. At those rates, small delays can quickly create large backlogs. More signal paths might be warranted not necessarily for capacity, but to maintain timely delivery by avoiding long queues of pending signals.

Overview of available signaling options

Figure 4 on page 100 is a diagram of a four-system sysplex in which signaling is implemented through CTC connections. The diagram shows the minimum required signaling connectivity for a four-system sysplex.

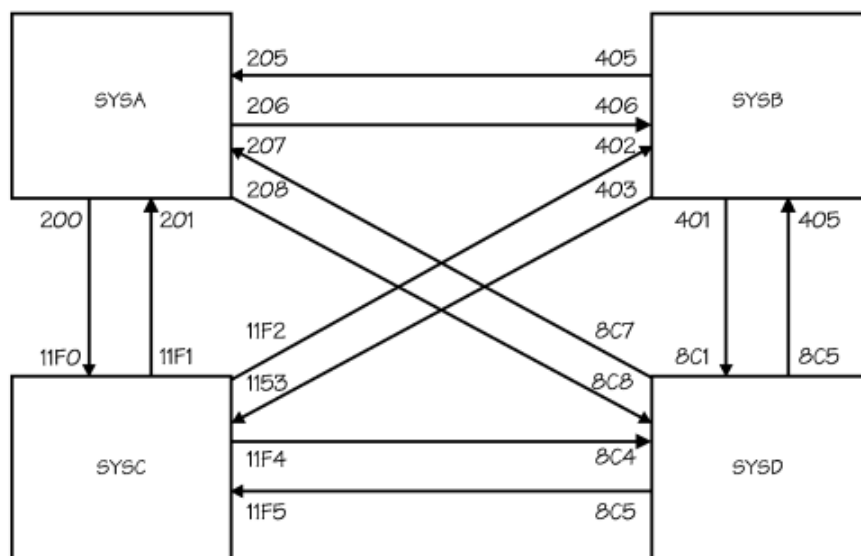


Figure 4. Signaling on a Four-System Sysplex, Implemented through CTC Connections

Figure 5 on page 101 shows the PATHOUT and PATHIN statements for SYSA coded for the configuration in Figure 4 on page 100. Note that the address of each device is specified for each DEVICE keyword.

```

/* Device that leads to SYSB. */
PATHOUT  DEVICE(206)

/* Device that leads to SYSC. */
PATHOUT  DEVICE(200)

/* Device that leads to SYSD. */
PATHOUT  DEVICE(208)

/* Device that comes from SYSB. */
PATHIN   DEVICE(205)

/* Device that comes from SYSC. */
PATHIN   DEVICE(201)

/* Device that comes from SYSD. */
PATHIN   DEVICE(207)

```

Figure 5. Example of SYSA's PATHIN and PATHOUT Specifications in COUPLExx

Figure 6 on page 101 is a diagram of a four-system sysplex in which signaling is implemented through a coupling facility list structure. The diagram shows the minimum required signaling connectivity for a four-system sysplex.

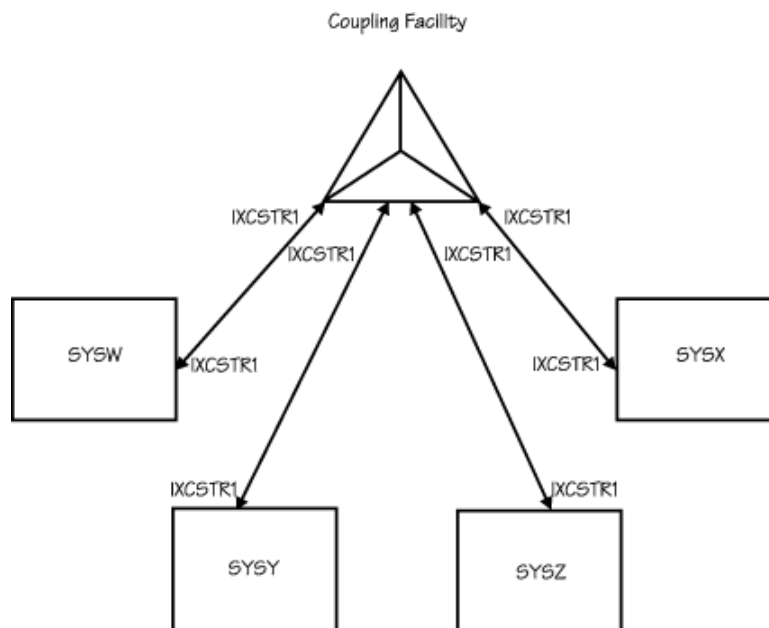


Figure 6. Signaling on a Four-System Sysplex, Implemented through a Coupling Facility List Structure

Figure 7 on page 101 shows the PATHIN and PATHOUT specifications in COUPLExx for SYSW shown in Figure 6 on page 101. Note that the same structure is specified for both PATHOUT and PATHIN and that the same specifications would be used on SYSX, SYSY, and SYSZ.

```

/* Structures that define paths to SYSX, SYSY, and SYSZ. */
PATHOUT  STRNAME(IXCSTR1)

/* Structures that define paths from SYSX, SYSY, and SYSZ. */
PATHIN   STRNAME(IXCSTR1)

```

Figure 7. Example of SYSW's PATHIN and PATHOUT Specifications in COUPLExx

Advantages of implementing signaling through list structures

Implementing signaling through coupling facility list structures provides significant advantages in the areas of systems management and recovery and, thus, provides enhanced availability for sysplex systems.

While a signaling path defined through CTC connections must be exclusively defined as either outbound or inbound, you can define a coupling facility list structure so that it is used for both outbound and inbound signaling paths, and MVS automatically establishes the paths. For example, if you define a list structure for outbound message traffic, MVS automatically establishes a signaling path with every other system that has the structure defined for inbound message traffic. Similarly, if you define a list structure for inbound traffic, MVS automatically establishes a signaling path with every other system that has the structure defined for outbound traffic.

Because a single list structure can be defined for both inbound and outbound traffic, you can use the same COUPLExx parmlib member for each system in the sysplex. A CTC connection, in contrast, cannot be defined to a single system as both inbound and outbound. Therefore, with CTCs, you must specify a unique COUPLExx member for each system in the sysplex or configure your systems so that they can use the same COUPLExx member by overspecifying the number of devices for signaling paths and tolerating failure messages related to unavailable devices and other configuration errors.

Implementing signaling through coupling facility list structures also enhances the recovery capability of signaling paths and reduces the amount of operator intervention required to run the sysplex.

Setting up outbound and inbound signaling paths

Whether you implement signaling through CTC connections or coupling facility list structures, you need to specify information in the COUPLExx parmlib member. After IPL, you can issue the SETXCF START,PATHOUT or SETXCF START,PATHIN command to specify outbound or inbound paths. To establish signaling through a coupling facility, you also need to define a coupling facility resource management (CFRM) policy.

Implementing Signaling through CTC Connections

Signaling paths implemented through CTC connections are one directional. That is, on each system, messages sent to other systems require an outbound path, and messages received from other systems require a separate inbound path.

To establish signaling paths through CTC devices, you specify their device numbers. For each system in the sysplex, you must specify all devices for outbound paths and inbound paths on the DEVICE keyword of the PATHOUT and PATHIN statements in COUPLExx.

When you issue SETXCF commands to start, stop, or modify signaling paths, you also need to specify the addresses of the devices.

Implementing signaling paths through coupling facility list structures

When you define signaling paths through coupling facility list structures, you define to MVS which list structures to use and how to use them (as pathin or pathout) and MVS creates the logical connections between systems that are using the structure for signaling.

To implement signaling paths through list structures, you need to set up a CFRM policy and define one or more signaling list structures for PATHIN and/or PATHOUT in COUPLExx. The name of each list structure you define for XCF signaling must begin with the letters IXC but not with IXCNP. The structure name prefix IXCNP is reserved for use by the XCF Note Pad Services. After IPL, you can use commands to define signaling list structures for PATHIN and/or PATHOUT.

Planning a CFRM policy for signaling

Chapter 4, “Managing coupling facility resources,” on page 41 provides considerations for setting up a CFRM policy. You format the couple data set that will contain the CFRM policy with the IXCLIDSU utility

program. You set up your CFRM policy using the administrative data utility (IXCMIAPU), which resides in SYS1.MIGLIB. See Chapter 12, “Administrative data utility,” on page 335.

When formatting the couple data set to contain the CFRM policy data, ensure that the value you specify for MAXSYSTEM matches the value for MAXSYSTEM that was used when the sysplex couple data set was formatted.

When coupling facility structures are used for XCF signaling and the MAXSYSTEM value specified for the CFRM couple data set is less than that of the sysplex couple data set, systems might not be able to join the sysplex. For example, if the only signaling connectivity is through coupling facility structures, MAXSYSTEM=16 is specified for the sysplex couple data set, and MAXSYSTEM=8 is specified for the CFRM couple data set, then at most eight systems will be allowed in the sysplex. (The limit might be less than eight, as described in section “Implications of the MAXSYSTEM value for the sysplex couple data set” on page 36.)

In the CFRM policy, you specify the name of the list structure, its size, and a preference list (a prioritized list of the coupling facilities on which you want the list structure allocated). Optionally, you can also specify an exclusion list — the names of any list structures you do not want allocated on the coupling facility the specified list structure is actually allocated on.

Avoiding a single point of failure

You should never have a single point of failure in your XCF signaling configuration. Loss of a single CTC link or single coupling facility structure could cause total loss of signaling connectivity between two or more systems in the sysplex. To avoid a single point of failure in the sysplex, specify the following in the CFRM policy:

- The names of at least two list structures for signaling
- For each list structure you define, the names of at least two coupling facilities in the preference list
- Exclusion lists, so that the list structures are allocated in different coupling facilities.

If you are especially concerned about availability, you can use CTC devices, as well as coupling facilities, for signaling because they provide a different transport mechanism with different hardware and software points of failure. However, this approach increases the complexity of systems management.

You also need to ensure that there is enough free space in the coupling facilities to allow MVS signaling services to recover from failures by rebuilding list structures. For high availability, you should never rebuild all the signaling structures in your XCF signaling configuration at the same time, unless there is adequate redundant CTC connectivity to compensate for the temporary unavailability of the signaling structures. While it is being rebuilt, an XCF signaling structure is quiesced and therefore is unusable until the rebuild process completes. Without redundant CTC connectivity during rebuild of XCF signaling structures, structure rebuild processing is significantly slowed down. This condition may result in one or more systems being removed from the sysplex when there is an active SFM policy with CONNFIL(YES) specified.

Note that the REBUILDPERCENT parameter does not apply to signaling structures.

Supporting dynamic alteration of signaling structures

An operator can alter (expand or contract) a signaling structure by issuing the SETXCF START,ALTER command. This might be a response, for example, to changes in the signaling workload, to changes in the number of systems using a structure for signaling, or to performance information.

To enable such a structure change to occur without disrupting any connecting systems, CFLEVEL=1 or higher must be specified in the preference list for the structure.

To accommodate expansion of the structure, it is recommended that you define the INITSIZE (initial size) and SIZE (maximum size) keywords for signaling structures so as to allow room for future expansion of the structure,

Determining the Size of a List Structure for Signaling

When you define a list structure, you must specify its size. (If the size you specify is too small, message IXC463I describes the minimum amount of space needed to allocate the structure.)

The size of a signaling list structure depends on the number of MVS systems in the sysplex, the level of the coupling facility, the level of MVS you are running, and the size, frequency, and performance requirements for the messages (signals) that the list structure is to accommodate.

IBM recommends that you use the CFSIZER tool to calculate the size of all your coupling facility structures.

COUPLExx specifications and commands

To implement signaling paths through list structures, you specify a list structure for use as PATHIN or PATHOUT and MVS automatically establishes a corresponding inbound or outbound path with all the systems that have specified that structure for pathout or pathin.

To implement signaling through a list structure, you need to specify (on the STRNAME keyword of the PATHOUT and PATHIN statements in COUPLExx) the name of the structure, defined in the CFRM policy, that holds the signaling path connections.

To implement signaling through a list structure, define the structure as both an outbound and an inbound path on every system connected to that facility. MVS creates one path in each direction for each path defined.

Handling signaling path failures

If a signaling path fails, MVS cannot use the path. Whether signaling is implemented through CTC connections or through list structures, if the underlying hardware fails, all the signaling paths using that hardware stop.

Through CTC devices

If signaling is implemented through CTC devices, the signaling paths cannot be recovered until the hardware recovers. When the hardware recovers, an operator can vary online an operative channel path or device, or the installation can reconnect an ESCON or FICON CTC device.

Through coupling facility list structures

If signaling is implemented through coupling facility list structures, and if a coupling facility that holds a list structure fails or if connectivity to a coupling facility that holds a list structure is lost, MVS can rebuild the list structure in another available coupling facility and reestablish signaling paths.

If the list structure itself fails, MVS can rebuild it in the same coupling facility or in another available coupling facility and then reestablish signaling paths.

Handling signaling connectivity failures

If a system loses connectivity to another, message IXC409D is issued, and the operator can try to re-establish connectivity or remove the system from the sysplex.

Or, if all systems are running OS/390 or MVS/ESA SP Version 5, the sysplex failure management policy can be used to isolate one or more disconnected systems from the sysplex. See [“Handling signaling connectivity failures” on page 183](#).

Note that if a structure used for signaling fails and there is no alternate signaling path, signaling connectivity is lost until the structure is rebuilt. During this time, if the sysplex failure management (SFM) policy is active, SFM might take action before the structure is rebuilt, and the sysplex could end up with only one surviving system. Therefore, signaling path redundancy is strongly recommended, particularly when SFM is active in the sysplex.

Deleting inoperative signaling paths

An inoperative signaling path remains defined to MVS. DISPLAY XCF lists the status of such a signaling path as INOPERATIVE. You can delete the definition of the inoperative path by issuing SETXCF STOP for the signaling path.

Communicating signaling path information to operators

It is recommended that you provide your operator with a diagram illustrating the layout of your signaling configuration. This diagram should include system names, device numbers assigned to inbound and outbound signaling paths or the names of list structures in which these paths are defined, and corresponding COUPLExx parmlib definitions. If coupling facilities are in use, the diagram should also include coupling facility-related information, such as their channel addresses. The operator will need this information if MVS is unable to establish connectivity between systems and requests operator intervention.

Planning the RETRY limit for paths

MVS maintains a retry count for each path in the sysplex. When an error occurs on a path, MVS retries the path. If the retry is unsuccessful, MVS adds one to the retry count; if the retry is successful, MVS subtracts one from the retry count. When the retry count matches the specified RETRY value, MVS stops the path.

You specify a path's retry limit on the RETRY keyword of the PATHIN and PATHOUT statements (or take the default specified on the RETRY keyword of the COUPLE statement). After IPL, you can specify retry limits using various SETXCF commands.

The retry limit for a signaling path defined through a list structure becomes the retry limit for every signaling path, in that direction (outbound or inbound), on the system that uses that structure for signaling.

When selecting a retry limit, consider that:

- A large retry limit delays the removal of an inoperative or error-prone path.
- A small retry limit hastens the removal of an inoperative or error-prone path. (Infrequent errors may be tolerable.)

For most installations, the default retry limit of 10 should be satisfactory. However, in a sysplex that includes MVS/ESA SP Version 4 systems (as well as SP Version 5 or OS/390 systems), IBM recommends increasing the retry value on the SP Version 4 systems to at least 15. This increase would allow the Version 4 systems to tolerate the additional retry attempts that later systems sometimes make when attempting to establish signaling connectivity.

Planning for optimum signaling performance

Optimal signaling performance for a sysplex implies that XCF groups in the sysplex have access to the signaling resources (signaling paths and message buffers) they need to communicate in a way that meets the overall communication goals for the sysplex.

IBM recommends using only defaults when initially setting up a sysplex and then changing values, assigning transport classes, and adding paths as you evaluate the operation of the sysplex.

To tune signaling performance, you need a way to separate XCF groups into different categories, as well as a way to associate these categories with specific signaling resources.

A transport class is MVS's way of enabling you to associate one or more XCF groups (based on similar signaling requirements) and then assign them signaling resources (signaling paths and message buffers).

To plan the signaling configuration, you need to:

- Identify the XCF groups in your sysplex and consider their signaling resource requirements.
- On each system, associate groups with similar signaling requirements and assign them to a transport class.

- Assign to each transport class the signaling paths and message buffers to meet these signaling requirements.

This information is presented as follows:

- [“XCF groups” on page 106](#)
- [“Planning the transport classes” on page 106](#)
- [“Planning the message buffer space” on page 112](#)

XCF groups

An XCF group is a set of related members that a multisystem application defines to XCF. A member is a specific function, or instance, of the application. A member resides on one system and can communicate with other members of the same group across the sysplex.

Communication between group members on different systems occurs over the signaling paths that connect the systems; on the same system, communication between group members occurs through local signaling services.

To prevent multisystem applications from interfering with one another, each XCF group name in the sysplex must be unique.

A multisystem application can use XCF group services (a set of authorized assembler macros) to obtain information about groups and members, define its group members to the sysplex, disassociate members from XCF services, and so forth. For more information about these macros, see [z/OS MVS Programming: Sysplex Services Guide](#).

XCF group names

To associate an XCF group and its members with a transport class, you need to know its XCF group name. XCF group names for some MVS components and IBM products are included in the discussion of [“Group Names and members for MVS components and subsystems” on page 32](#).

You also need to know the XCF group names for any other multisystem application programs that communicate among members on various systems. An application programmer who codes a multisystem application that uses signaling services should give you the application's XCF group name so that you can designate transport class(es), if needed, for the group's use. Otherwise, for a list of the XCF groups defined on the system, you can issue the DISPLAY XCF,GROUP command or run the RMF XCF Activity Report.

Message Characteristics for XCF Groups

To define suitable transport classes for an XCF group, you must know the lengths, frequency and performance requirements for messages sent by the group.

- For message information on MVS groups, see [“Message information for some MVS groups” on page 409](#).
- [“Coupling facility structures for IBM products” on page 96](#) provides reference sources for IBM product applications.
- An application programmer who codes a multisystem application that uses signaling services should provide you with the lengths, frequency, and performance requirements for messages sent by members of the group.

The RMF XCF Activity Report is another source of information about messages for the groups in the system.

Planning the transport classes

In general, transport classes are not needed if the XCF function switch XTCSIZE is enabled on every system in the sysplex. Transport class definitions direct the XCF component of z/OS to classify XCF message traffic based on the length of the message, the name of the sender's XCF group, or both. When an XCF exploiter sends a message, the group name and message length are used to determine

the most suitable transport class. The message is then transferred to the target system by using the message buffers and signal paths assigned to the selected transport class. Thus, transport classes force XCF to use a particular subset of signal resources when sending messages. In effect, different classes of messages can be isolated from one another. For example, you could define transport classes so one set of signal paths is used for small messages and a different set of signal paths is used for large messages. A transport class is said to be a size-only transport class if XCF groups are not assigned to the class. That is, when defining the transport class, the GROUP keyword is either omitted or coded as GROUP(UNDESIG).

Historically, size-only transport classes helped optimize the XCF signal service. Originally, transport classes were needed to help minimize transfer times for small messages. Intermixing large messages with small ones might noticeably elongate the transfer of the small messages, which in turn might degrade sysplex performance. Over the years, the technology improved. Intermixing the transfer of different size messages no longer has a significant impact on sysplex performance. In recent times, the transport classes were primarily used to avoid the overhead (and delays) associated with changing the message size that is supported by a given signal path, and to make more effective use of the available message buffer space. Today, if every system runs with the XCF function switch XTCSIZE enabled, the sysplex is self-optimizing and size-only transport class definitions are no longer needed. The signal resources are said to be “XCF Managed”. An XCF-Managed signal path can send any size signal with no additional overhead since the path is always tuned to support the maximum signal size. The use of outbound message buffer space is optimal since a best fit buffer is used for every message. However, if the XTCSIZE function is disabled or the sysplex has systems that do not support the XTCSIZE function, size-only transport classes might still be needed.

Size-only transport classes always influence XCF signal path selection. If XTCSIZE is disabled or the target system does not support the XTCSIZE function, traditional transport class processing applies. Only the signal paths assigned to the best-fit transport class are selected. If XTCSIZE is enabled and the target system supports XTCSIZE, not-busy signal paths that are assigned to the best-fit transport class are given preference. If those signal paths are all busy, any path from any size-only class can be selected. This behavior wherein existing transport classes influence the selection of XCF-Managed signal paths was intended as an interim aid to help avoid significant changes in signal path usage as systems were upgraded to z/OS releases with XTCSIZE support. In the long term, the size-only classes should be eliminated because this preference for a subset of signal paths might not provide optimal signal performance. For example, the signal paths that are assigned to the best-fit transport class might not provide the fastest transfer times. Eliminating the size-only transport classes eliminates the skewed path selection and allows XCF to choose signal paths based solely on performance.

In general, using transport classes to classify signal traffic based on XCF groups is not recommended for production environments. The signal resources for such classes are available only to the XCF groups assigned to the class, and the groups that are assigned to the class are restricted to using only the signal resources from the classes to which they are assigned. In effect, the signal traffic of the designated XCF groups is isolated from the signal traffic that is generated by other groups. Typically, one might define such classes for one of two reasons: either to provide favored treatment to a group whose traffic is deemed critical, or to isolate an ill-behaved group so it cannot impede the signal traffic of others. In practice, the use of transport classes for these purposes has proven to be problematic or unnecessary:

- Using a transport class to isolate an ill-behaved group.

From an XCF signaling perspective, an ill-behaved group is one whose members fail to process signals in a timely manner. The untimely processing delays the recycling of XCF message buffers, which in turn can impede signal transfer over the signal paths, which can lead to queuing and “no buffer” conditions on the sending systems. Dedicating signal resources to such a group limits these ill effects to the group itself. In practice, this might not be practical. The primary difficulty is identifying the ill-behaved groups. You might not discover the ill-behaved group until after it impacts your sysplex. In some cases, you might know which exploiter is potentially ill-behaved but be unable to predict its dynamically generated group name. Historically, ill-behaved members tended to impact XCF signal transfer more significantly than they do with z/OS releases that support XTCSIZE. Various enhancements over the years now make it possible for XCF to maintain signal throughput despite the presence of ill-behaved members. Thus, the need to define transport classes to mitigate the impact of ill-behaved members has largely been eliminated.

- Using a transport class to isolate a favored group.

Historically, it was thought such transport classes would ensure that messages sent by a critical group might be transferred without interference from messages sent by other groups. The message traffic for a critical group tends to be either high frequency or sporadic. However, the very nature of the transport class isolation often proved problematic. Signal path performance issues sometimes induced impactful delays for the high frequency sender that might have been avoided if XCF was free to choose better performing paths outside the group's class. For the sporadic sender, the XCF working set of buffers was often empty when it came time to send the signals. While trying to create a message buffer for the request, system issues sometimes induced impactful delays that might have been avoided if XCF was free to choose an existing buffer from a transport class whose working set was nonempty.

Thus, you generally create greater risk by dedicating signal resources to the favored group than by letting it compete with other groups for signal resources. You can overcome the potential “bad path” issues by providing a set of signal paths for the group that is as comprehensive and diverse as the one provided for a transport class defined for small messages. If the group sends different size signals, you might also need to optimize signal performance for the group by assigning it to multiple classes so its signals can be partitioned by size. Either way, there will typically be an excess of unused signal capacity being dedicated to the group. A single group seldom generates enough message traffic to warrant the number of signal paths needed to overcome the potential “bad path” issues. Furthermore, the underutilized signal paths further compound the empty working set issues.

Transport classes to isolate the signal traffic for a particular group can be of use to testers. Restricting a group's signal traffic to a particular set of signal paths can make it easier to manage the test, drive various scenarios, and better understand the signal behavior of the assigned group.

Note: The XTCSIZE function does not apply to a transport class that has been assigned a group (other than UNDESIG).

Defining Transport Classes

Transport classes are defined independently for each system in the sysplex by using the CLASS keyword of the CLASSDEF statement or after IPL, using the SETXCF START,CLASSDEF command. [Figure 8 on page 108](#) shows the syntax of the statement.

```
[CLASSDEF
    CLASS(class-name)
    [CLASSLEN(class-length)
    [GROUP(group-name[,group-name]...)]
    [MAXMSG(max-messages)]
```

Figure 8. Syntax for CLASSDEF Statement of COUPLExx parmlib Member

On a system, each transport class must have a unique name. The class name is used in system commands and shown in display output and reports.

For some workloads, the same transport class definition is suitable for all systems in the sysplex. However, it is not unusual for there to be significant variation in the signal traffic among the systems in the sysplex. This variation can make it difficult to create optimal transport class definitions. For example, a transport class for a given message size might have significant volume on certain systems and virtually no use on others. For the systems with little use, the signal paths assigned to the class are underutilized. Those signal paths are unavailable for use by messages in other classes. Creating a unique set of transport class definitions for those systems might help overcome that issue. However, doing so increases complexity and the amount of manual tuning effort required. Nor does it necessarily resolve the issue since the signal traffic patterns need not be static. The dynamics of the sysplex are such that the nature of the signal traffic between the systems can vary over time, both concerning volume and message size.

Assigning XCF groups to a transport class

By explicitly assigning an XCF group to a transport class, you give the group priority access to the signaling resources (signaling paths and message buffer space) of the transport class. All groups assigned to a transport class have equal access to the signaling resources of that class.

Assign one or more XCF groups to a transport class using the GROUP keyword of the CLASSDEF statement or, after IPL, using the SETXCF START,CLASSDEF command or the SETXCF MODIFY,CLASSDEF command.

Undesignated Groups

If you do not specify the GROUP keyword on a CLASSDEF statement, no groups are explicitly assigned to the transport class. A group that you do not assign to a transport class is called an undesignated group. The set of all undesignated groups is collectively referred to on the CLASSDEF keyword and system commands by the pseudo-group name UNDESIG. The undesignated groups are implicitly assigned to the transport classes whose CLASSDEF statement did not specify the GROUP keyword. The undesignated groups can be explicitly assigned to a transport class by specifying GROUP(UNDESIG) on the CLASSDEF statement. After IPL, additional groups (including the UNDESIG set of groups) can be explicitly assigned to a transport class using the SETXCF START,CLASSDEF command or the SETXCF MODIFY,CLASSDEF command.

On systems that support the XTCSIZE function, XCF defines a pseudo-transport class named _XCFMGD. When the XTCSIZE optional XCF FUNCTION is ENABLED, undesignated groups are eligible for assignment to the _XCFMGD pseudo-transport class. Transport classes to which only undesignated groups are assigned are not directly used when XTCSIZE is ENABLED and communicating with a system that supports the XTCSIZE function. Instead, resources associated with those transport classes will be reassigned to the _XCFMGD pseudo-transport class.

Using the default transport class

The default transport class (named DEFAULT) is implicitly defined by the system using the MAXMSG and CLASSLEN specifications from the COUPLE statement in the COUPLExx parmlib member (or if unspecified, the system default for the respective keyword). Changing those values on the COUPLE statement is one way to alter the definition of the default class. Alternatively, you can explicitly define the default class by coding a CLASSDEF statement in the COUPLExx parmlib member with CLASS(DEFAULT) and your own MAXMSG and CLASSLEN specifications. After IPL, you can use the SETXCF MODIFY,CLASSDEF,CLASS=DEFAULT command to alter the definition (such changes will persist only for the life of the IPL).

Note: If you create your own definition for the DEFAULT class, IBM suggests that you omit the GROUP keyword to make it a size-only class. There must be at least one size-only transport class defined to derive benefit from enabling the XTCSIZE function.

Outbound signal paths are assigned to the default transport class if their defining PATHOUT statement in the COUPLExx parmlib member omits the CLASS keyword or CLASS(DEFAULT) is specified. If the PATHOUT statement omits the MAXMSG keyword, the MAXMSG value for the outbound path is set to the MAXMSG value of the transport class to which it is assigned. As for any installation defined transport class, the total amount of buffer space available to the default transport class (for a given target system) is the MAXMSG value for the transport class plus the sum of the MAXMSG values for each of the class's outbound paths that are connected to the given system. If you omit MAXMSG on the defining PATHOUT statements, the amount of buffer space for a given target system is simply $M \times (N+1)$ where M is the MAXMSG value for the class and N is the number of outbound signal paths in the class to that system.

For installations running with the XCF function switch XTCSIZE enabled on all systems in the sysplex, only the default class is needed. No other size-only transport class should be defined. In the general case, enabling XTCSIZE causes all resources for size-only transport classes to be logically reassigned to an XCF defined pseudo-transport class named _XCFMGD (provided the target system supports the XTCSIZE function). In the specific case where the default transport class is the only such transport class, the XCF-Managed pseudo-class _XCFMGD will look much like the default class. However, there are differences between _XCFMGD and DEFAULT (or any other size-only transport class):

- The signal buffer space available to _XCFMGD will typically equal the space available to the default class (or all the size-only classes combined). It can be larger if XCF deems that buffer supply inadequate. An adequate buffer supply requires at least 2000 K of buffer space per XCF-Managed signal path. In addition, the base MAXMSG value for _XCFMGD will be the larger of 2000 and the MAXMSG value that is specified on the COUPLE statement in the COUPLExx parmlib member.
- A best fit buffer is always used for _XCFMGD. A best fit buffer is the smallest message buffer that can be used for a given message. The buffers that are used for a traditional transport class will typically be at least as large as the best fit buffer for its class length (CLASSLEN). Thus, a traditional transport class buffer could be bigger than necessary.
- The signal paths for _XCFMGD run at the maximum signal size. Thus, any size signal can be sent over any XCF-Managed signal path with no additional overhead. The signal paths for a traditional transport class typically run at the “class size”. The class size is the length of the biggest message that fits in a buffer that best fits the class length. If a message exceeds the class size, its transfer is delayed if XCF needs to increase the message size currently supported by the selected signal path.

If the XTCSIZE function is enabled on all systems in the sysplex and you choose to define size-only transport classes other than DEFAULT, be aware that those classes skew XCF-Managed path selection. XCF will first try to use a not-busy path per traditional transport class selection rules. Since XCF signal paths are typically not busy, this behavior can lead to suboptimal signaling performance since the signal paths in the selected class might not be the best performing (though this is generally no worse than what the traditional transport classes would have achieved). If DEFAULT is the only class, XCF has but one class to consider, and XCF path selection will naturally gravitate toward using the best performing signal paths overall. As you transition from a sysplex with multiple size-only transport classes without XTCSIZE support to one where XTCSIZE is everywhere enabled, the skewed path selection helps minimize changes in behavior. The skewing might also prove useful if you have reasons for wanting XCF to give preference to certain signal paths despite the potential for suboptimal signaling performance. For example, you might want to direct XCF signal traffic toward or away from a given coupling facility. However, this “direction” is not absolute since XCF looks to use other XCF-Managed signal paths if the ones in the selected size-only class are all busy. If you want to use traditional size-only transport classes to force XCF to choose specific paths, you must disable the XTCSIZE function switch to revert to traditional transport class processing. You then bear responsibility for configuring, monitoring, and tuning those transport classes to ensure good signal performance.

If DEFAULT is the only transport class and you choose to disable the XTCSIZE function, be aware of the following consequences:

- If DEFAULT is configured for the smallest buffer size, all messages are sent using best-fit buffers. This is equivalent to _XCFMGD and provides for maximum buffer space capacity. However, unlike XCF Managed, the signal paths are normally tuned to that smallest class size. Larger messages incur more overhead and delay if the selected path needs to increase its current maximum signal size to accommodate the message. Since the preponderance of signal traffic is small messages, XCF tuning algorithms will likely change the path size back to the class size as frequently as once per second. Each path size change creates more control signals and a short quiesce period during which the subject inbound path will not receive signals. In the worst case, a given signal path might undergo nearly 20 such size changes per second. The XCF signal configuration and dynamics of the system determine how impactful these size changes are to signal performance. The potential for significant impact increases as the signal rate increases. At high signal rates, small delays can lead to significant queuing.
- If DEFAULT is configured for the largest class size, all signal paths are tuned to that size. This is equivalent to _XCFMGD. Every message can be sent without changing the signal size that is supported by the path. However, unlike XCF Managed, messages would typically use the largest buffer size. This provides the least efficient use of the available buffer space, which can lead to send requests being rejected due to “no buffer” conditions. Not all exploiters deal gracefully with “no buffer” conditions, so you might be tempted to increase the allowable buffer space to avoid them. But doing so increases the risk of an outage if the limit goes beyond a level of storage consumption that is safe for the system. Typically, the number of buffers that are needed to maintain signal flow is small, but the appropriate number ultimately depends on the dynamics of the sysplex. In general, the most critical factor is the responsiveness of the target system. The risk of consuming all the allowable buffer space increases as the signal rate increases or the target system becomes less responsive.

- If DEFAULT is configured for a class size between the smallest and largest, there will be a mix of the two behaviors. As the class size increases, the need to adjust the path size decreases but the use of the available buffer space becomes less efficient. Given the dynamics of the sysplex, it is generally difficult to choose a single class size that always works well for all target systems.

Before the XTCSIZE function, user-defined transport classes were generally needed to mitigate these issues. Though seldom perfect, the transport classes helped increase the effective number of outbound message buffers while reducing the need for signal paths to change their message size. Thus, the ill effects of inefficient buffer space utilization and size changes might be kept to levels that typically would not noticeably degrade signal performance. However, enabling XTCSIZE sysplex-wide is now the preferred mechanism for achieving that goal. XTCSIZE always uses best-fit buffers and need not change the path size when sending a message. Optimal signal performance can be achieved without transport classes.

Specifying the lengths of messages for the transport class

Specify the length of the messages for which the signaling service should optimize in a transport class on the CLASSLEN keyword of the CLASSDEF statement or, after IPL, on the SETXCF START,CLASSDEF or the SETXCF MODIFY,CLASSDEF command.

The class length determines the size of the XCF message buffers used to hold messages sent in that transport class. z/OS selects the smallest buffer that will hold messages of the size specified on the CLASSLEN keyword. In general, specify a class length equal to the length of the messages most frequently sent by the one or more groups in the transport class.

Specify the length of the messages (CLASSLEN) for a transport class based on the one or more groups you assign to the transport class. Therefore, you must know the characteristics of the messages the groups send.

Selecting an appropriate class length involves making compromises between storage use and signaling performance. If messages are usually shorter than the message buffer, storage is wasted; if longer, performance is degraded.

A message whose length does not exceed the buffer length receives the best performance the signaling service can provide. Messages longer than the buffer length might require additional processing, such as the preparation of additional message buffer space or the sending of additional signals to deliver the oversized messages.

If there are many messages longer than the buffer length and the message traffic patterns warrant it, the signaling service dynamically increases the size of the buffers in the class to avoid the additional processing overhead to deliver these messages. This dynamic adjustment of buffer sizes means that specifying a small class length does not necessarily mean that larger signals will suffer performance degradation. However, if increasing the buffer size would cause the signaling service to exceed the maximum message buffer space allowed for messages in a particular transport class for a particular receiving system, these adjustments are not made. (See [“Planning the message buffer space”](#) on page 112.)

Planning the signaling paths for the transport class

You assign outbound signaling paths or local message buffers to a transport class by using the CLASS keyword of the PATHOUT or LOCALMSG statements or after IPL, using the SETXCF (START or MODIFY) path command. Inbound signaling paths are not directly assigned to a transport class. However, an inbound signal path effectively inherits the transport class definition of the outbound signal path to which it is connected. The inherited class length specification might impact the choice of MAXMSG value for the inbound signal path. [Figure 9 on page 112](#) shows the syntax.

```
[PATHIN
    {DEVICE(device-number[,device-number]...) }
    {STRNAME(strname[,strname]...) }
    [MAXMSG(max-messages)]
    [RETRY(retry-limit) ]
[PATHOUT
    {DEVICE(device-number[,device-number]...) }
    {STRNAME(strname[,strname]...) }
    [MAXMSG(max-messages)]
    [RETRY(retry-limit) ]
    [CLASS(class-name)]
[LOCALMSG
    MAXMSG(max-messages)
    [CLASS(class-name)]
```

Figure 9. Syntax for PATHIN, PATHOUT, LOCALMSG on COUPLExx parmlib Member

If you implement signaling through a list structure, you use the STRNAME keyword of the PATHIN or PATHOUT statement to specify one or more names of one or more list structures you want to use for PATHIN or PATHOUT on that system. If you assign a structure to a transport class, every outbound signaling path that MVS starts through that structure to other systems are also assigned to that transport class. (If the list structure is not explicitly assigned to a transport class, it is implicitly assigned to the default transport class.)

If you implement signaling through CTCs, you can assign each CTC to a transport class. Because a CTC connects to a specific system, you have more flexibility to add paths between systems within a class.

Assign signaling paths to a transport class based on the amount of message traffic that you expect the class to handle. One signaling path might be sufficient for most transport classes; however, assign two (or more) paths if you want a redundant path available to the transport class or require additional capacity.

An XCF group usually uses only the signaling paths that are associated with the transport classes to which the group is assigned. However, if there are no paths available to a group in any of its assigned transport classes, messages in the group are routed to other paths.

Failure to assign a signaling path to a transport class can degrade the performance of the signaling service for XCF groups that are assigned to that class. In this case, the groups compete for the signaling resources of transport classes that are assigned to other XCF groups.

Planning the message buffer space

Messages sent and received through sysplex signaling services occupy a message buffer in central storage while being processed. These buffers are allocated as needed to support the message traffic load.

At times, runaway applications, non-operational remote systems, or structure rebuilds can cause message traffic to back up to the point that the amount of storage acquired for signaling would degrade the rest of the system.

The MAXMSG parameter allows you to limit the total amount of storage (message buffer space) available to a transport class or signaling path.

Specifying Message Buffer Space

You specify the maximum message buffer space on the MAXMSG keyword of the COUPLE, CLASSDEF, PATHOUT, PATHIN, and LOCALMSG statements in the COUPLExx parmlib member. If you implement signaling through list structures, the MAXMSG value takes effect for all paths on that system that are started through that structure. You can modify these values using the SETXCF MODIFY command.

Table 8 on page 113 summarizes the MAXMSG values that the system uses if MAXMSG is not specified on the statement:

Table 8. MAXMSG Values on COUPLE Statements

Statement	MAXMSG Value If Not Coded on Statement
COUPLE	2000K
CLASSDEF	MAXMSG value, if coded on COUPLE statement; otherwise, 2000K
PATHOUT	MAXMSG value, if coded on CLASSDEF statement for the transport class. If MAXMSG is not coded on CLASSDEF statement, then MAXMSG value for COUPLE statement Otherwise, 2000K
PATHIN	MAXMSG value, if coded on COUPLE statement; otherwise, 2000K
LOCALMSG	None; that is, zero

Kinds of message buffers

There are three kinds of message buffers. Outbound and inbound message buffers allow members of groups to communicate with other systems. Outbound message buffers are used to send messages to another system; inbound message buffers are used to receive messages from another system. Local message buffers are used to send and receive messages within the same system.

The various types of message buffers are isolated from one another so that the supply of one type does not affect the supply of any other type. Message buffers used to communicate with one system are isolated from those used to communicate with other systems. Thus, exhausting the supply of message buffers used to communicate with one system does not impact communication with any other system.

Outbound and Local message Buffers

On any given system, XCF maintains one or more buffer pools for each target system. The number of such pools is determined by your transport class definitions. The buffer pools for any given target system are independent of the buffer pools for any other system. If a target system has multiple buffer pools, those pools are also independent of one another. When an exploiter issues a send request, XCF selects one of the buffer pools and obtains a message buffer. A local message buffer is used if the target resides on the same system as the sender. If the target resides on some other system, an outbound buffer is used, and a signal path is selected to transmit the message. This creates a temporary bind between the outbound buffer and the signal path. The bind persists until either (a) the signal is known to have been successfully transferred to the target system, or (b) an alternative path is chosen to do the transfer because the selected path failed (the bind then moves to the alternative path). Each outbound signal path always has a small set of message buffers so bound. Thus, some amount of outbound buffer space will always be in use. The remainder of the allotted space is available to service newly arriving send requests. The amount of outbound buffer space needed to maintain signal flow is largely determined by the ability of the system to accomplish signal transfer, which is largely determined by the ability of the target system to receive messages in a timely manner. The amount of local buffer space needed to maintain signal flow is largely determined by the time it takes to deliver messages to the local target members.

Note that the outbound paths do not have individual buffer pools. Sometimes this is a point of confusion because a MAXMSG value can be specified for each outbound signal path and the seemingly persistent set of buffers that are bound to the signal path makes it look like the path is maintaining its own buffer supply. However, all buffer management is done at the scope of a target system buffer pool. The MAXMSG value of an outbound path is that path's contribution to the total buffer space available in the buffer pool with which the path is associated. The path's MAXMSG contribution of buffer space does not "belong" to the path, it belongs to the buffer pool. The message in any given buffer can be sent via any given signal path. The amount of buffer space bound to a signal path for transfer could exceed the MAXMSG specification of the path. That's because the only bearing the outbound path MAXMSG value has on buffer management is to increase the buffer space available to the buffer pool.

Figure 10 on page 114 shows SYSA's message buffers with the local and outbound buffers partitioned by class (TC1, TC2) and the outbound buffers partitioned by system (SYSB, SYSC, SYSD). Typically, a message is transmitted to the target system via some outbound signal path that is assigned to the same class as the outbound buffer. In that sense, the outbound paths for a given class and target system share the outbound buffers from that system's transport class buffer pool. Figure 10 suggests that SYSA's

inbound message buffers are partitioned by system only (SYSB, SYSC, SYSD). In fact, every inbound path has a dedicated pool of inbound buffers that are not shared with any other path. Since a given inbound signal path receives from exactly one system, we can think of the collection of inbound paths that are connected to a given sending system as providing a buffer pool partitioned by system. However, since the inbound buffer pools are managed on a per path basis, it is better to think of the inbound buffers as being partitioned by inbound signal path.

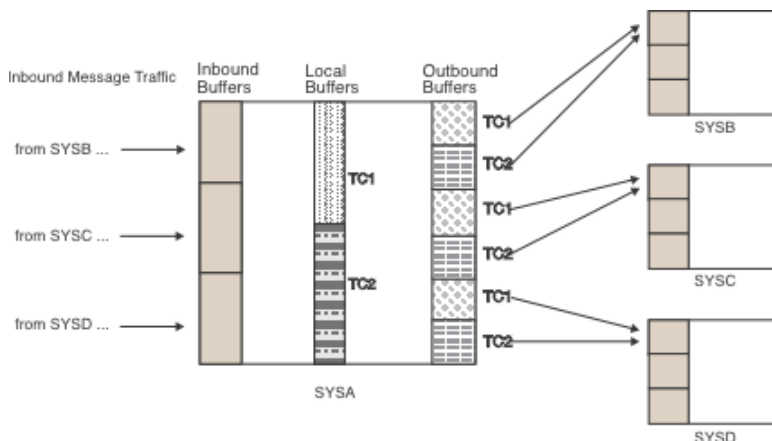


Figure 10. SYSA with Inbound, Local, and Outbound message Buffers

The message buffer space available to outbound messages in a transport class is the sum of the message buffer space contributed by the transport class plus the message buffer space contributed by each outbound signaling path in that class. If the available message buffer space is not adequate, then messages are rejected.

The message buffer space available to outbound messages in the _XCFMGD pseudo-transport class is generally the sum of the message buffer space available to the associated size-only transport classes. Note that if that sum is small, the system can choose a reasonable minimum amount of message buffer space.

Because the message buffer space available for communicating with another system depends on the contribution of message buffer space by the outbound signaling paths that are connected to that system, the loss of a signaling path can cause a reduction in the available buffer space. Moreover, the operator can use commands to change the amount of message buffer space contributed by a signaling path or a transport class definition. Therefore, it is possible for the amount of message buffer space in use to exceed the buffer space limit. In this situation, MVS rejects message requests until the buffer space in use is consistent with the new buffer space limit.

The total message buffer space available to local messages in a transport class is the sum of the message buffer space contributed by the transport class (specified or defaulted on the MAXMSG parameter of the CLASSDEF or COUPLE statement) plus any message buffer space that is specified on the LOCALMSG statement.

The message buffer space available to local messages in the _XCFMGD pseudo-transport class is generally the sum of the message buffer space available for local messages in all associated size-only transport classes. Note that if that sum is small, the system can choose a reasonable minimum amount of message buffer space.

You want to specify sufficient message buffer space to support the message traffic for each system in each transport class that is used for delivering messages to that system. When specifying buffer space for a transport class, consider the size and frequency of the messages, as well as the performance of the signaling paths and systems involved in the message transfer.

Inbound message buffers

Each inbound signal path has a dedicated pool of inbound message buffers. The buffer pool for any given inbound signal path is independent of the buffer pool for any other inbound path. Subject to the MAXMSG limit for the given inbound signal path, XCF dynamically grows and shrinks the buffer pool as needed to

maintain signal flow while conserving buffer space. Since an inbound CTC path always has I/O pending, there will always be some amount of buffer space in use. An inbound list path initiates I/O on demand. Depending on path selection by the sending system, there might be times when an inbound list path does not have any buffers in use. Beyond the immediate use for I/O, the consumed amount of inbound buffer space is largely determined by the amount of time it takes to deliver and process the messages that have been received.

The message buffer space available to inbound messages is specified or defaulted on the MAXMSG parameter of the PATHIN statement. If you implement signaling with CTCs, you can specify the message buffer space for each inbound signaling path. If you implement signaling through list structures, the message buffer space you define for the structure on a system takes effect for all the paths started by that system through that list structure.

When specifying message buffer space for an inbound signaling path, consider the size and frequency of the messages to be received on that path. You want to provide enough message buffer space to allow the inbound signaling path to receive messages without causing messages to back up on the sending system. To determine the message size, note the class length of the sending system's transport class.

If the message buffer space for an inbound signaling path is too small to support a message sent from a remote system, MVS exceeds the buffer space limit to receive the message.

Calculating message buffer space

In general, you should not be concerned with the buffer space required for an individual message. However, you must ensure that the message buffer space is large enough to hold a message of the length specified (on the CLASSLEN keyword) for the transport class.

However, when the message traffic in your sysplex requires you to limit the number of message buffers, you do need to know how much buffer space is required to support a particular message length. For detailed information on this topic, see [“Calculating message buffer space — An example” on page 410](#).

When you define a list structure for signaling, more than one signaling path might be started. Therefore, to compute the message buffer space required by the system, you must anticipate the maximum number of signaling paths that might be started through the list structure.

Summary

To make decisions on maximum message buffer space, you need to understand the peak message traffic loads, distributions of messages among the systems in the sysplex, the sizes of those messages, and the consequences of having a message request rejected due to a lack of buffer space. A measurement tool such as RMF is helpful in determining message traffic patterns and loads. For information about RMF reports, see [Chapter 7, “Tuning a sysplex,” on page 149](#).

Adding and deleting signaling paths

Use the SETXCF START and SETXCF STOP commands to add and delete signaling paths.

Adding a signaling path

To add a signaling path, issue the SETXCF START command.

Through CTC devices

To add a signaling path defined through CTCs, the device must be unallocated before you issue the SETXCF START command.

When you add an outbound path on the sending system, also add its associated inbound path on the receiving system.

- To ensure that the device is not allocated, issue:

```
DISPLAY  U,CTC,ALLOC
```

- To add an inbound or outbound signaling path, issue:

```
SETXCF  START,PATHIN,DEVICE=indevnum  
SETXCF  START,PATHOUT,DEVICE=outdevnum
```

Use the MAXMSG, RETRY, and CLASS keywords as needed.

Through coupling facility list structures

To add a signaling path defined through a list structure, the structure must be defined in the active CFRM policy.

To add a signaling path defined in the CFRM policy, issue:

```
SETXCF  START,PATHIN,STRNAME=stname  
SETXCF  START,PATHOUT,STRNAME=stname
```

Deleting a signaling path

To delete a signaling path, issue the SETXCF STOP command.

Through CTC devices

When you delete an outbound path on the sending system, also delete its associated inbound path on the receiving system so the path is available for other users.

- To ensure that the device for an inbound or outbound path is defined to XCF, issue:

```
DISPLAY  XCF,PATHIN,DEVICE=(indevnum)  
DISPLAY  XCF,PATHOUT,DEVICE=(outdevnum)
```

- To delete an inbound or outbound signaling path, issue:

```
SETXCF  STOP,PATHIN,DEVICE=indevnum  
SETXCF  STOP,PATHOUT,DEVICE=outdevnum
```

Use UNCOND=YES when you need to terminate an active stop request that may have failed and initiate a new stop request.

When you stop a path, MVS unallocates the path. The path remains online to the system for other users.

Through coupling facility list structures

To ensure that a structure is defined to MVS for signaling, issue:

```
DISPLAY  XCF,PATHOUT,STRNAME=stname  
DISPLAY  XCF,PATHIN,STRNAME=stname
```

To ensure that the structure is defined in the active CFRM policy, issue:

```
DISPLAY  XCF,STRNAME=stname
```

To delete an inbound or outbound signaling path defined through a list structure, issue:

```
SETXCF  STOP,PATHOUT,STRNAME=stname  
SETXCF  STOP,PATHIN,STRNAME=stname
```

Note that these commands stop all the (inbound or outbound) paths through the structure on the system, as well as the corresponding (outbound or inbound) paths on the systems to which the paths connect. Stop PATHOUT before PATHIN to avoid delays due to additional outbound signal activity on the structure that would need to be resolved prior to the structure being deleted.

Considerations

When deleting signaling paths, consider the following:

- The operator cannot stop the last active signaling path to or from another system in the sysplex. MVS requires full connectivity (at least one outbound and one inbound signaling path) between each of pair of systems in a sysplex.
- When an MVS system in the sysplex is running as a guest under VM, exercise care when changing paths. If a device is detached and linked via CP commands while it is online to MVS, MVS boxes the device, which makes it unusable until the guest MVS system is re-IPLed.

Sample signaling scenarios

The following topic contains examples of various signaling configurations.

1. [“Example 1 — Using COUPLExx defaults” on page 117](#)

This example is a configuration of two systems that shows only the required statements and parameters and takes the system defaults for all others.

2. [“Example 2 — Creating a transport class for large messages” on page 119](#)

This example is a configuration of two systems that shows the required statements and parameters and those statements and parameters that you can use to partition message traffic based on message length.

3. [“Example 3 — Creating a transport class for a group” on page 121](#)

This example is a configuration of two systems that shows the required statements and parameters and those statements and parameters that you can use to partition message traffic based on XCF group.

4. [“Example 4 — Defining signaling paths through the coupling facility” on page 123](#)

This example is a configuration of two systems that uses a signaling configuration consisting entirely of coupling facility signaling structures.

5. [“Example 5 — Signaling through a coupling facility and CTC connections” on page 125](#)

This example is a configuration of two systems that uses a signaling configuration consisting of both coupling facility signaling structures and CTC devices.

To determine how to calculate the device addresses specified on the PATHIN and PATHOUT statements and shown on the example diagrams, see:

- *3088 Product Description*
- *Introducing Enterprise Systems Connection*

Example 1 — Using COUPLExx defaults

This example illustrates a sysplex of two systems in which all defaults in the COUPLExx parmlib member are taken. Only the required statements and parameters are shown.

IBM recommends that you use only the defaults when you initially set up a sysplex. You can then examine the XCF activity reports produced by RMF to determine whether a significant number of messages are too big to fit the default buffer sizes. Only in this case should you consider defining transport classes to accommodate larger messages. See [“Example 2 — Creating a transport class for large messages” on page 119](#).

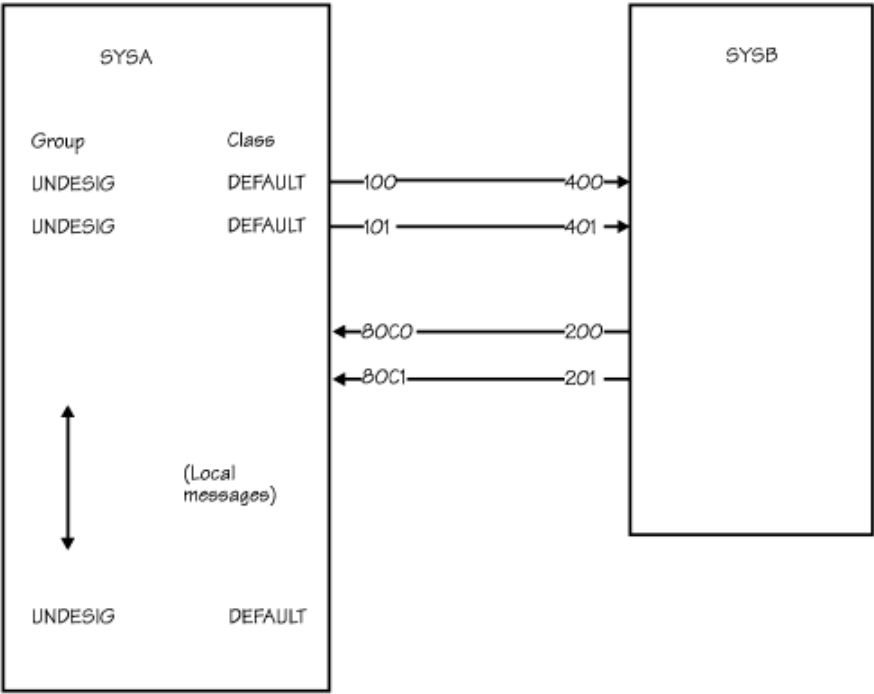
Note that there is no capability to control the class length for the _XCFMGD pseudo-transport class and whether messages fit or not in the RMF report can be ignored for the _XCFMGD pseudo-transport class.

In this scenario, PLEXOF2 ([Figure 11 on page 118](#)) consists of two MVS systems (SYSA and SYSB) on two processors. In this example, all defaults are used, and no transport classes are defined. Therefore, MVS creates only the default transport class (named DEFAULT) and assigns all groups (which are collectively

called UNDESIG) to the DEFAULT class. Also, MVS assigns all signaling paths to the DEFAULT class, which means that all groups share all paths equally.

Example 1 – Diagram of PLEXOF2

Figure 11 on page 118 is a logical diagram of the example sysplex, PLEXOF2.



Note: In this example, UNDESIG refers collectively to all groups not assigned to a transport class. In this case, every group on the system uses the signalling resources assigned to the transport class DEFAULT.

Figure 11. Example 1 – Diagram of PLEXOF2

Example 1 – SYSA COUPLExx parmlib member

Figure 12 on page 118 shows an example of the COUPLExx parmlib member for SYSA in PLEXOF2.

```
COUPLE    SYSPLEX(PLEXOF2)
          PCOUPLE(PLEXOF2.CDS1)
          ACOUPLE(PLEXOF2.CDS2)

/* Devices that lead to SYSB. */
PATHOUT  DEVICE(100,101)

/* Devices that come from SYSB. */
PATHIN   DEVICE(80C0,80C1)
```

Figure 12. Example 1 – SYSA's COUPLExx Parmlib Member

The COUPLE statement

- SYSPLEX(PLEXOF2) - identifies the name of the sysplex, PLEXOF2.

- PCOUPLE(PLEXOF2.CDS1) - identifies the primary sysplex couple data set.
- ACOUPLE(PLEXOF2.CDS2) - identifies the alternate sysplex couple data set. (Although optional, an alternate sysplex couple data set is recommended.)

/* Devices that lead to SYSB. */

- PATHOUT DEVICE(100,101) - defines outbound signaling paths 100 and 101.
MVS assigns the outbound paths to the DEFAULT transport class.

/* Devices that come from SYSB. */

- PATHIN DEVICE(80C0,80C1) - defines inbound signaling paths 80C0 and 80C1.

Example 1 — SYSB COUPLExx parmlib member

The COUPLExx parmlib member for SYSB in PLEXOF2 is shown in [Figure 13 on page 119](#).

```
COUPLE    SYSPLEX(PLEXOF2)
          PCOUPLE(PLEXOF2.CDS1)
          ACOUPLE(PLEXOF2.CDS2)

/* Devices that lead to SYSA. */
PATHOUT   DEVICE(200,201)

/* Devices that come from SYSA. */
PATHIN    DEVICE(400,401)
```

Figure 13. Example of SYSB's COUPLExx Parmlib Member

Example 2 — Creating a transport class for large messages

This example illustrates a sysplex of two systems in which the transport class (BIG) is defined to handle messages of up to 40,000 bytes and all other defaults in the COUPLExx parmlib member are taken.

In this scenario, the sysplex PLEXOF2B consists of two MVS systems (SYSA and SYSB) on two processors. In the sysplex, the length of messages that are sent by members of various groups vary widely and thus, messages require special consideration:

- All groups on SYSA have members that send small messages (up to 1,000 bytes) to their group members on SYSB.
- Some groups on SYSA have members that send longer messages (up to 40,000 bytes) to their group members on SYSB.

Historically, a large amount of signal traffic with varying lengths necessitated the use of transport classes to improve signal performance. Partitioning messages with size-only transport classes effectively increased the number of message buffers available for use and eliminated some (perhaps all) of the delay that is associated with dynamically reconfiguring signal paths to transport different size messages. This example illustrates how such size-only transport classes might be created. However, if the XCF function switch XTCSIZE is enabled on all systems in the sysplex, these size-only transport classes are no longer needed. Indeed, for optimal signal performance, they should not be defined at all. Only the default transport class should be used. With XTCSIZE everywhere enabled, the system automatically maximizes the number of possible outbound message buffers and eliminates the need for signal paths to change buffer sizes. The system implicitly achieves optimal signal performance without the need for the installation to define, monitor, or tune transport class definitions.

Example 2 — Diagram of PLEXOF2B

[Figure 14 on page 120](#) shows a logical diagram of the example sysplex named PLEXOF2B.

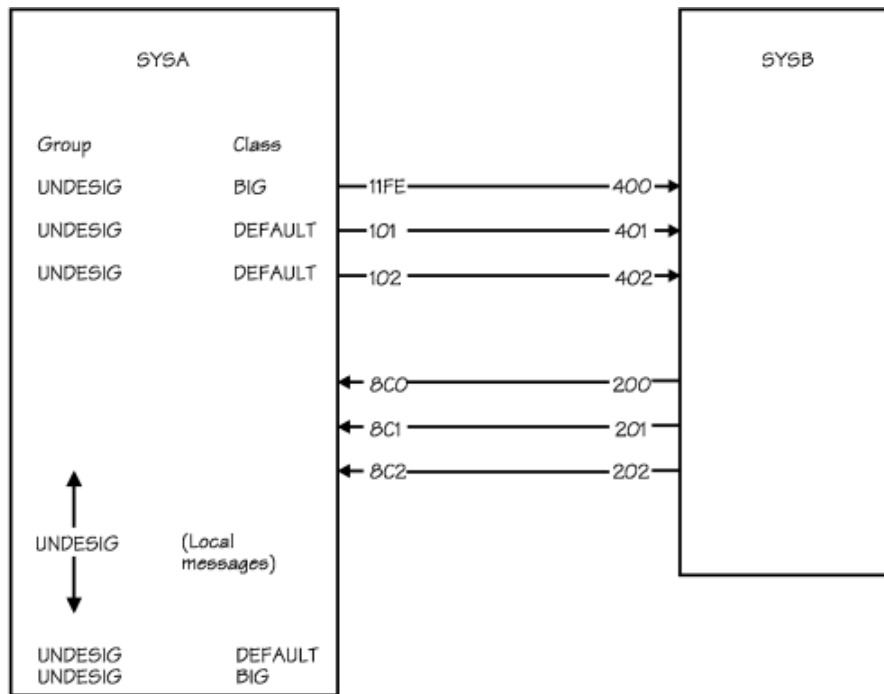


Figure 14. Example 2 — Diagram of PLEXOF2B

Example 2 — SYSA COUPLExx parmlib member

Figure 15 on page 120 shows an example of a COUPLExx parmlib member for SYSA in sysplex PLEXOF2B. Assume that defaults are taken for any keywords not coded.

```
COUPLE    SYSPLEX(PLEXOF2B)
          PCOUPLE(PLEXOF2B.CDS1)
          ACOUPLE(PLEXOF2B.CDS2)

/* Transport classes. */
CLASSDEF CLASS(BIG) CLASSLEN(40000)

/* Devices that lead to SYSB. */
PATHOUT  DEVICE(11FE) CLASS(BIG)
PATHOUT  DEVICE(101,102)

/* Devices that come from SYSB. */
PATHIN   DEVICE(8C1,8C2)
PATHIN   DEVICE(8C0) MAXMSG(2000)
```

Figure 15. Example 2 — SYSA's COUPLExx Parmlib Member

The COUPLE statement

- **SYSPLEX(PLEXOF2B)** - identifies the name of the sysplex, PLEXOF2B. The sysplex name specified here must match the sysplex name used in the Format Utility job to format the couple data sets specified by ACOUPLE and PCOUPLE.
- **PCOUPLE(PLEXOF2B.CDS1)** - identifies the primary sysplex couple data set.
- **ACOUPLE(PLEXOF2B.CDS2)** - identifies the alternate couple data set.

The CLASSDEF statement

- **CLASS(BIG)** - specifies BIG as the name of the transport class.

- CLASSLEN(40000) - specifies that messages to be handled by this class are up to 40,000+ bytes long. This specification causes MVS to optimize its processing for messages of up to (slightly more than) 40,000 bytes.

MVS defines the DEFAULT transport class automatically; it is not shown. MVS uses a class length of 956 for class DEFAULT; this length is sufficient for the small messages. Because no group is specifically assigned to any transport class, UNDESIG refers to all groups.

The CLASSDEF statement (for class BIG and class DEFAULT) partitions messages into two categories. Messages that are less than or equal to 956 bytes are sent using resources assigned to the transport class, DEFAULT. Messages between 956 bytes and 40,000+ bytes are sent using resources assigned to the BIG transport class. Note that messages longer than 40,000+ bytes are subject to performance degradation.

With the XTCSIZE switch ENABLED, DEFAULT and BIG resources may be reassigned to the _XCFMGD pseudo-transport class, which is sufficient for all message sizes.

/* Devices that lead to SYSB. */

- PATHOUT DEVICE(11FE) CLASS(BIG) - assigns outbound signaling path 11FE to class BIG. Outbound messages (the long 40,000-byte messages) for all groups are sent on this path.
- PATHOUT DEVICE(101,102) - defines outbound signaling paths 101 and 102. Outbound messages (the short messages) for all groups are sent on these paths.

/* Devices that come from SYSB. */

- PATHIN DEVICE(8C1,8C2) - defines inbound signaling paths 8C1, and 8C2.
- PATHIN DEVICE(8C0) - defines inbound signaling path 8C0. This path is defined to handle big messages.

Example 3 — Creating a transport class for a group

This example illustrates a sysplex of two systems in which a new application is assigned to its own transport class, TCNEW, and all other defaults in the COUPLExx parmlib member are taken. This ensures that message traffic for the new application is sent on its own dedicated signaling path and does not interfere with other multisystem applications.

In this scenario, the sysplex PLEXOF2A consists of two MVS systems (SYSA and SYSB) on two processors. There is one group, NEWAPP, in the sysplex that requires special consideration. NEWAPP is assigned to its own transport class (TCNEW) in the COUPLExx parmlib member on each system in the sysplex.

Members of other groups on this system share signaling resources in the DEFAULT transport class.

You might consider using this configuration when you plan to use an application and want to ensure other multisystem applications are not impacted by its use of signaling resources. For example, you might be unsure how a new application behaves in the sysplex. Alternatively, you might have evidence that the application's use of the signal service is detrimental to other users.

Typically, transport classes that partition message traffic by group are not needed. In general, XCF can manage the use of the signal service by an "ill behaved" group to avoid detrimental impact to other groups. Note that if you choose to define such a class, you must ensure that the class has the resources that are needed to meet the signal performance needs of the isolated group. In this example, there is but one signal path that is provided for the group. That one path can provide the necessary capacity when operating normally. However, if the performance of that signal path is degraded, the transport class definition forces the group's message traffic to use the poorly performing path. If the signal path fails and is removed from service, there is a loss of transport class connectivity. The group's signals will then be transferred via signal paths from other classes. At that point, the message traffic is no longer partitioned by group and the intended isolation is not achieved.

Example 3 — Diagram of PLEXOF2A

The following figure shows a logical diagram of the example sysplex named PLEXOF2A.

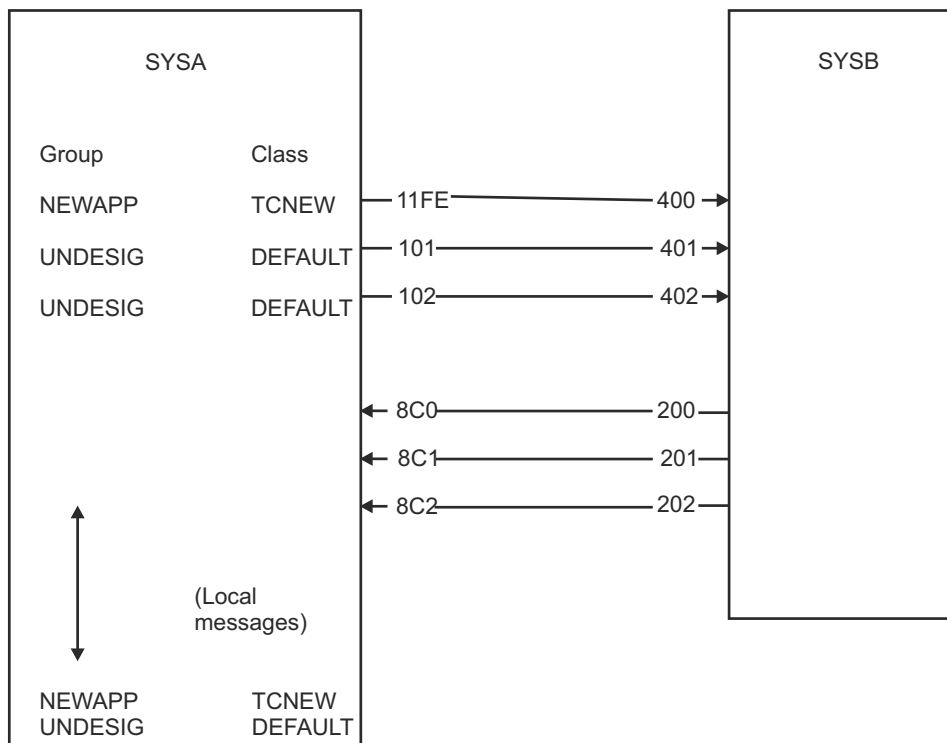


Figure 16. Example 3 — Diagram of PLEXOF2A

Example 3 — SYSA COUPLExx parmlib member

Figure 17 on page 122 shows an example of a COUPLExx parmlib member for SYSA in PLEXOF2A. Assume that defaults are taken for any keywords not coded.

```
COUPLE    SYSPLEX(PLEXOF2A)
          PCOUPLE(PLEXOF2A.CDS1)
          ACOUPLE(PLEXOF2A.CDS2)

/* Transport class. */
CLASSDEF CLASS(TCNEW) GROUP(NEWAPP)

/* Devices that lead to SYSB. */
PATHOUT  DEVICE(11FE) CLASS(TCNEW)
PATHOUT  DEVICE(101,102)

/* Devices that come from SYSB. */
PATHIN   DEVICE(8C0,8C1,8C2)
```

Figure 17. Example of SYSA's COUPLExx Parmlib Member

The COUPLE statement

- SYSPLEX(PLEXOF2A) - identifies the name of the sysplex.
- PCOUPLE(PLEXOF2A.CDS1) - identifies the primary sysplex couple data set.
- ACOUPLE(PLEXOF2A.CDS2) - identifies the alternate sysplex couple data set.

The CLASSDEF statement

- CLASS(TCNEW) - specifies TCNEW as the name of the transport class.
- GROUP(NEWAPP) - assigns group NEWAPP to transport class TCNEW.

The defaults are taken for CLASSLEN and MAXMSG parameters.

MVS defines the DEFAULT transport class automatically. In this case, because the group NEWAPP is assigned to a transport class, UNDESIG refers to all groups except NEWAPP.

/* Devices that lead to SYSB. */

- PATHOUT DEVICE(11FE) CLASS(TCNEW) - assigns outbound signaling path 11FE to class TCNEW. Outbound messages for group NEWAPP are sent on this path.
- PATHOUT DEVICE(101,102) - defines outbound signaling paths 101 and 102. MVS assigns the outbound paths to the DEFAULT class. Outbound messages for all groups, other than NEWAPP, are sent on these paths.

/* Devices that come from SYSB. */

- PATHIN DEVICE(8C0,8C1,8C2) - defines inbound signaling paths 8C0, 8C1, and 8C2.

In the example, the COUPLExx parmlib member on SYSB could also define a TCNEW transport class, assign the NEWAPP group to TCNEW, and assign an outbound path (such as 200) to class TCNEW.

Example 4 – Defining signaling paths through the coupling facility

This example illustrates a sysplex of two systems whose signaling paths are defined exclusively through the coupling facility. In this example, the group, named NEWAPP, is assigned to its own transport class, and all other COUPLExx defaults are taken. This ensures that message traffic for the members of the NEWAPP group is sent on its own dedicated signaling structure and does not compete with other multisystem applications.

In this scenario, the sysplex named PLEXOF2C consists of two MVS systems (SYSA and SYSB) on two processors.

One group, NEWAPP, in the sysplex requires special consideration. NEWAPP is assigned to its own transport class (TCNEW) in the COUPLExx parmlib member on each system in the sysplex.

Members of other groups on this system share signaling resources in the DEFAULT transport class.

You might consider using this configuration when you plan to use an application for which you do not want other multisystem applications competing for signaling resources.

Example 4 – Diagram of PLEXOF2C

Figure 18 on page 124 shows a logical diagram of the example sysplex named PLEXOF2C.

Note that in this example, the two signaling structures are allocated in two different coupling facilities (as specified in the coupling facility resource management (CFRM) policy). This would normally be done for high availability in the event of a coupling facility failure or loss of connectivity to a coupling facility, and is the recommended configuration. However, this separation is not required; the two signaling structures can be placed in the same coupling facility.

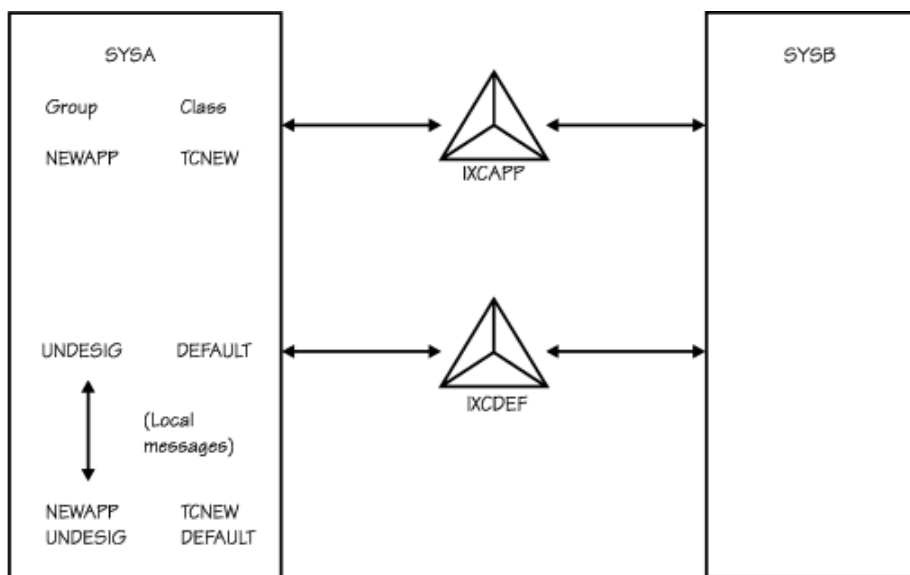


Figure 18. Example 4 – Diagram of PLEXOF2C

Example 4 – SYSA COUPLExx parmlib member

An example of the COUPLExx parmlib member for SYSA in PLEXOF2C is shown in Figure 19 on page 124. (Note that the same COUPLExx parmlib member statements could be used on **any** system in the sysplex, since the PATHIN and PATHOUT structure definitions could be the same on any system.) Assume that defaults are taken for any keywords not coded.

```
COUPLE    SYSPLEX(PLEXOF2C)
          PCOUPLE(PLEXOF2C.CDS1)
          ACOUPLE(PLEXOF2C.CDS2)

/* Policy type */
DATA TYPE(CFRM)    PCOUPLE(PLEXOF2C.CFRM1)
                  ACOUPLE(PLEXOF2C.CFRM2)

/* Transport class */
CLASSDEF CLASS(TCNEW)
          GROUP(NEWAPP)

/* Paths out */
PATHOUT STRNAME(IXCAPP) CLASS(TCNEW)
PATHOUT STRNAME(IXCDEF)

/* Paths in */
PATHIN
STRNAME(IXCAPP,IXCDEF)
```

Figure 19. Example 4 – SYSA's COUPLExx Parmlib Member

The COUPLE statement

- SYSPLEX(PLEXOF2C) - identifies the name of the sysplex, PLEXOF2C. The sysplex name specified here must match the sysplex name used in the Format Utility job to format the sysplex couple data sets specified by ACOUPLE and PCOUPLE.
- PCOUPLE(PLEXOF2C.CDS1) - identifies the primary sysplex couple data set.
- ACOUPLE(PLEXOF2C.CDS2) - identifies the alternate sysplex couple data set.

The DATA Statement

- TYPE(CFRM) - specifies that the CFRM policy is to be used.
- PCOUPLE(PLEXOF2C.CFRM1) - identifies the primary CFRM couple data set.

- ACOUPLE(PLEXOF2C.CFRM2) - identifies the alternate CFRM couple data set.

The CLASSDEF statement

- CLASS(TCNEW) - specifies TCNEW as the name of the transport class.
- GROUP(NEWAPP) - assigns group NEWAPP to transport class TCNEW.

The defaults are taken for CLASSLEN and MAXMSG parameters.

MVS defines the transport class DEFAULT automatically. In this case, because the group NEWAPP is assigned to a transport class, UNDESIG refers to all groups except NEWAPP. (The set of all unassigned groups are collectively referred to on the CLASSDEF keyword as UNDESIG).

/* Paths out. */

- PATHOUT STRNAME(IXCAPP) CLASS(TCNEW) - defines signaling structure IXCAPP for outbound use. Outbound signaling paths will be established with every other system that defines this structure for inbound use. These outbound signaling paths are assigned to transport class TCNEW. Outbound messages for group NEWAPP are sent via this signaling structure.
- PATHOUT STRNAME(IXCDEF) - defines signaling structure IXCDEF for outbound use. Outbound signaling paths will be established with every other system that defines this structure for inbound use. Since no transport class is explicitly specified for this structure, MVS assigns the outbound paths to the class DEFAULT. Outbound messages for all groups, other than SYSAPP, are sent via this signaling structure.

/* Paths in. */

- PATHIN STRNAME(IXCDEF,IXCAPP) - defines signaling structures IXCDEF and IXCAPP for inbound use. Inbound signaling paths will be established with every other system that defines this structure for outbound use.

Note: Unlike a CTC device, a signaling structure can be defined for simultaneous inbound and outbound use.

In the example, the COUPLExx parmlib member on SYSB could also define a TCNEW transport class, assign the NEWAPP group to TCNEW, and assign structure IXCAPP as pathout from SYSB. In such a configuration, all NEWAPP signaling traffic in the sysplex would be preferentially sent via the IXCAPP structure, and all other signaling traffic in the sysplex would be preferentially sent via the IXCDEF structure.

Example 5 — Signaling through a coupling facility and CTC connections

This example illustrates a sysplex of two systems whose signaling paths are defined both via the coupling facility **and** CTC devices. In this example, the group (named SYSAPP) is assigned to its own transport class, and all other COUPLExx defaults are taken. This ensures that message traffic for the members of the group NEWAPP is sent on its own dedicated signaling paths and does not compete with other multisystem applications.

Note that because more signaling paths are defined, the configuration in this example provides higher availability than the example shown in [“Example 4 — Defining signaling paths through the coupling facility” on page 123](#).

In this scenario, the sysplex PLEXOF2D consists of two MVS systems (SYSA and SYSB) on two processors.

There is one group, NEWAPP, in the sysplex that requires special consideration. NEWAPP is assigned to its own transport class (TCNEW) in the COUPLExx parmlib member on each system in the sysplex.

Members of other groups on this system share signaling resources in the DEFAULT transport class.

Consider using this configuration when you plan to use an application for which you do not want other multisystem applications competing for signaling resources.

Example 5 – Diagram of PLEXOF2D

Figure 20 on page 126 shows a logical diagram of the example sysplex named PLEXOF2D. Note that in this example, the two signaling structures are allocated in two different coupling facilities (as determined by the CFRM policy). This is done to maintain availability in case of a coupling facility failure or loss of connectivity to a coupling facility, and is the recommended configuration. However, this separation is not required; the two signaling structures can be placed in the same coupling facility. The example also shows three CTC connections between SYSA and SYSB.

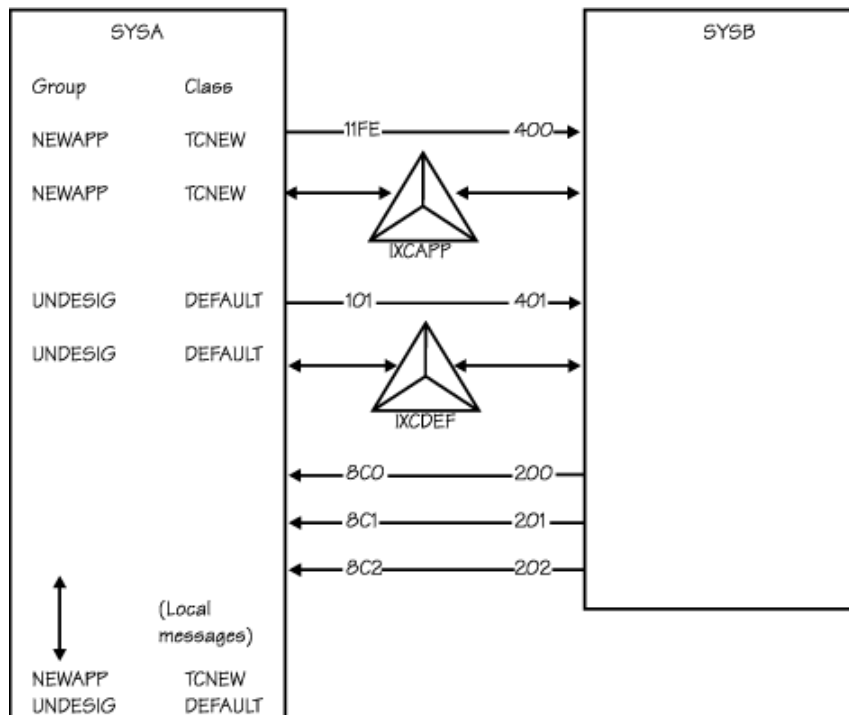


Figure 20. Example 5 – Diagram of PLEXOF2D

Example 5 – SYSA COUPLExx parmlib member

An example of the COUPLExx parmlib member for SYSA in PLEXOF2D is shown in the following figure. Assume that defaults are taken for any keywords not coded.

```
COUPLE    SYSPLEX(PLEXOF2D)
          PCOUPLE(PLEXOF2D.CDS1)
          ACOUPLE(PLEXOF2D.CDS2)

/* Policy Type */
DATA      TYPE(CFRM)
          PCOUPLE(PLEXOF2D.CFRM1)
          ACOUPLE(PLEXOF2D.CFRM2)

/* Transport class */
CLASSDEF  CLASS(TCNEW) GROUP(NEWAPP)

/* Paths out */
PATHOUT   STRNAME(IXCAPP) CLASS(TCNEW)
PATHOUT   DEVICE(11FE) CLASS(TCNEW)
PATHOUT   STRNAME(IXCDEF)
PATHOUT   DEVICE(101)

/* Paths in */
PATHIN    STRNAME(IXCAPP,IXCDEF)
PATHIN    DEVICE(8C0,8C1,8C2)
```

Figure 21. Example of SYSA's COUPLExx Parmlib Member

The COUPLE statement

- SYSPLEX(PLEXOF2D) - identifies the name of the sysplex, PLEXOF2D. The sysplex name specified here must match the sysplex name used in the Format Utility job to format the couple data sets specified by ACOUPLE and PCOUPLE.
- PCOUPLE(PLEXOF2D.CDS1) - identifies the primary sysplex couple data set.
- ACOUPLE(PLEXOF2D.CDS2) - identifies the alternate sysplex couple data set.

The DATA Statement

- TYPE(CFRM) - specifies that the CFRM policy is to be used.
- PCOUPLE(PLEXOF2D.CFRM1) - identifies the primary CFRM couple data set.
- ACOUPLE(PLEXOF2D.CFRM2) - identifies the alternate CFRM couple data set.

The CLASSDEF statement

- CLASS(TCNEW) - specifies TCNEW as the name of the transport class.
- GROUP(NEWAPP) - assigns group NEWAPP to transport class TCNEW.

The defaults are taken for CLASSLEN and MAXMSG parameters.

MVS defines the DEFAULT transport class automatically. In this case, because the group NEWAPP is assigned to a transport class, UNDESIG refers to all groups except NEWAPP.

/* Paths out. */

- PATHOUT STRNAME(IXCAPP) CLASS(TCNEW) - defines signaling structure IXCAPP for outbound traffic. Outbound signaling paths will be established with every other system that defines IXCAPP for inbound traffic. These outbound signaling paths are assigned to transport class TCNEW. Outbound messages for group NEWAPP are sent via IXCAPP.
- PATHOUT DEVICE(11FE) CLASS(TCNEW) - assigns outbound signaling path 11FE to class TCNEW. Outbound messages for group SYSAPP are sent on this path.
- PATHOUT STRNAME(IXCDEF) - defines signaling structure IXCDEF for outbound use. Outbound signaling paths will be established with every other system that defines IXCDEF for inbound use. Since no transport class is explicitly specified for this structure, MVS assigns the outbound paths to the DEFAULT class. Outbound messages for all groups, other than NEWAPP, are sent via IXCDEF
- PATHOUT DEVICE(101) - defines outbound signaling path 101. MVS assigns the outbound path to the DEFAULT class. Outbound messages for all groups, other than SYSAPP, are sent on this path.

/* Paths in. */

- PATHIN STRNAME(IXCDEF,IXCAPP) - defines signaling structures IXCDEF and IXCAPP for inbound use. Inbound signaling paths will be established with every other system that defines these structures for outbound use.

Note: Unlike a CTC device, a signaling structure can be defined for simultaneous inbound and outbound use.

- PATHIN DEVICE(8C0,8C1,8C2) - defines inbound signaling paths 8C0, 8C1, and 8C2.

In the example, the COUPLExx parmlib member on SYSB could also define a TCNEW transport class, assign the NEWAPP group to TCNEW, and assign structure IXCNEW and/or a signaling CTC device (such as 200) as pathout from SYSB to class TCNEW.

Handling signalling path problems

Use the DISPLAY XCF command and Resource Measurement Facility (RMF) reports to obtain information about paths. In response to the DISPLAY XCF,PATHIN or PATHOUT command, XCF issues message IXC356I, which indicates information about the paths, including the status of signalling paths as:

- **STARTING:** The path is being verified.
- **RESTARTING:** The path is being restarted; for example, after a failure.
- **LINKING:** The path is waiting for a connection to another system.
- **WORKING:** The path is available for message transfer.
- **QUIESCING:** The path is being quiesced.
- **STOPPING:** The path is being removed from service.
- **STOPFAILED:** The path requires intervention.
- **INOPERATIVE:** The path is unusable but remains defined to XCF.
- **REBUILDING:** Rebuild has been initiated for this signalling path list structure.
- **STALL-IOPND:** The path has pending I/O that is not completing in a timely manner.
- **STALL-INOP :** The path is not capable of transferring signals. The inbound side does not have any buffers available for receiving signals.
- **STALL-SS?:** The path is not capable of transferring signals and is being monitored for potential sympathy sickness impact. The inbound side is experiencing no buffer conditions that is caused by one or more stalled XCF group members. The outbound side avoids using the path until the problem is resolved.
- **STALL-SS :** The path is not capable of transferring signals. The sending side is suffering from sympathy sickness since signals cannot be transferred.

When a coupling facility structure is being used for signalling, the operator can issue other DISPLAY commands for additional information about the status of the signalling structures:

- **DISPLAY CF** provides information about space utilization within a coupling facility.
- **DISPLAY XCF,CF** identifies structures allocated within a coupling facility and identifies each system with connectivity to the coupling facility.
- **DISPLAY XCF,STRUCTURE** provides information about the attributes of a structure, such as its size, maximum number of supported connections, or disposition. **DISPLAY XCF,STRUCTURE** also indicates the status of a structure, including whether it is in the rebuilding process.

Also, the operator can enter the **DISPLAY U,CTC** command to obtain information about the CTC devices.

The following topics describe signalling path problems.

- [“Loss of some signalling paths between systems” on page 129](#)
- [“Loss of all signalling paths between some systems” on page 130](#)
- [“Loss of all signalling paths between all systems” on page 130](#)

Operations Note: When signalling paths are down, XCF cannot communicate between systems. MVS component exploiters, such as global resource serialization, and other multisystem applications might be forced to stop work in the sysplex. Therefore, plan to respond to signalling path problems in a timely manner.

Problems with signalling path definitions

The following problems can occur with signalling path definitions at IPL time:

- Incorrect signalling path definitions
- Not enough signalling path definitions
- Unable to establish connectivity
- Incorrect coupling facility structure definitions

Incorrect signalling path definitions

When the operator IPLs a system into a multisystem sysplex, XCF on each of the systems in the sysplex checks the signalling path definitions in its COUPLExx parmlib member.

If one of the following reasons are found, XCF on the initializing system or on an existing system issues message IXC451I. The reasons detected by XCF on an existing system are:

- **NOT SAME SYSPLEX:** A signalling path on the initializing system is connected to a signalling path on this (an existing) system that is not using the same sysplex name as the initializing system. All paths must connect systems within the same sysplex.
- **COUPLING DATASETS DIFFER:** A signalling path on the initializing system is connected to a signalling path on this (an existing) system that is not using the same sysplex couple data set as the initializing system. All paths must connect systems that use the same couple data set.

The reason detected by XCF on the initializing system is:

- **CIRCULAR PATH:** An outbound signalling path on the initializing system is connected to an inbound signalling path that is also on the initializing system. This is not allowed.

Also message IXC458I might indicate the device address of a signalling path that has been stopped or has failed.

Not enough signalling path definitions

If there are not sufficient signalling paths defined to provide at least one inbound and one outbound path between each of the systems in the sysplex, XCF on the initializing system issues message IXC453I to indicate that signalling path connectivity could not be established.

The operator is then prompted for a different COUPLExx parmlib member.

Check also for message IXC451I on other systems in the sysplex. This message indicates incorrect signalling path definitions that can result in message IXC453I.

Unable to Establish Connectivity

If signalling paths cannot be established to provide at least one inbound and one outbound path between each of the systems in the sysplex, XCF on the initializing system issues message IXC454I, which indicates that signalling connectivity cannot be established. XCF then issues message IXC455D and requests the operator to respond with:

R

To be prompted for a different COUPLExx parmlib member.

INTERVAL=NNN

To specify the number of seconds that XCF is to continue checking for signalling connectivity to other systems in the sysplex. Use INTERVAL when one or more other systems, shown in message IXC454I, are slow to respond to the requests of the initializing system to establish connectivity.

Message IXC451I might indicate incorrect signalling path definitions that can result in message IXC454I.

Loss of some signalling paths between systems

If one or more paths are lost between systems and there is at least one outbound and one inbound path between all systems, XCF uses the remaining paths for message traffic. XCF issues message IXC458I or IXC459I for most problems involving signalling paths.

Some reasons why XCF stops a path include:

- The operator has stopped the path.
- The retry limit of the path is reached.
- The system is removed from the sysplex.
- A subchannel command has failed.
- XCF does not have exclusive use of the path.

Note that a loss of paths can cause performance problems with message traffic because more group members are competing for the remaining paths. Additional outbound and inbound paths can be assigned to the systems via the SETXCF START,PATHIN and PATHOUT commands. If a path is listed

as INOPERATIVE, in some cases XCF can automatically restart the signalling path. Thus, the operator does not need to redefine the path on the SETXCF START command.

Loss of all signalling paths between some systems

If one or more paths are lost between systems and there is not at least one outbound and one inbound path between two of the systems in the sysplex, XCF issues message IXC409D. Message IXC409D requests the operator to either retry the path(s) or to remove the system from the sysplex.

If a retry of the paths is not successful, the operator responds with the name of the system to be removed from the sysplex. To avoid removing a system from the sysplex, the operator may be able to specify additional outbound and inbound paths via the SETXCF START,PATHIN and PATHOUT commands.

A loss of paths between two systems can be caused by a channel failure on one of the systems when there is not a second channel providing signalling paths between the systems.

To rejoin the sysplex, a system must be reinitialized with sufficient signalling paths between the systems.

Loss of all signalling paths between all systems

If all paths are lost between all systems, XCF issues message IXC409D. Message IXC409D requests the operator to either retry the path or paths or to remove the system from the sysplex.

If a retry of the paths is not successful, the operator responds with the name of the system to be removed from the sysplex. This is repeated until there is only one system left in the sysplex. To avoid removing a system from the sysplex, the operator may be able to specify additional outbound and inbound paths via the SETXCF START,PATHIN and PATHOUT commands.

When coupling facility structures are being used for signalling, path loss can occur because of a coupling facility failure when there are no backup paths available, either on another coupling facility or ESCON Director. Path loss also can occur because of a coupling facility failure when it is not possible to rebuild the structure in another coupling facility.

To rejoin the sysplex, a system must be reinitialized with sufficient signalling paths between the systems.

Signaling sympathy sickness

signaling sympathy sickness can occur to a sending system when one or more members become stalled on the target system. The stalled target member fails to recycle its inbound buffers and therefore cannot receive signals properly. If signals are not being received, outbound signal buffers on the sending system remain in use. If the situation persists, XCF on the sending system will eventually run out of the outbound buffers and any further IXCMSGO requests will be rejected.

When sympathy sickness is detected, message IXC631I is issued by the system causing the problem to indicate that a local stalled XCF member appears to be causing sympathy sickness on some other system in the sysplex. This message is displayed once for each system impacted by the stalled member.

The system causing the sympathy sickness will also issue one instance of message IXC640E to indicate that a problem exists and to describe a method of resolution. However, both message IXC631I and message IXC640E may not be displayed if consoles on the sick system has been impacted and never processed the messages.

Message IXC440E is issued by the impacted system to point out which system was causing the sympathy sickness. Note that even if message IXC640E is not displayed on the faulty system, message IXC440E can still be displayed on the impacted system. Thus it is recommended that operational procedures or system automation be updated to deal with message IXC440E. See *z/OS MVS System Messages, Vol 10 (IXC-IZP)* for more information on these messages.

You can use the DISPLAY XCF,GROUP command to investigate stalled members and signaling sympathy sickness if manual intervention is required. You can also use the MEMSTALLTIME parameter in the SFM policy to allow SFM to take automatic action to alleviate signaling sympathy sickness. See [“Handling signaling sympathy sickness” on page 185](#) for more information about defining an SFM policy for sympathy sickness.

WSC flash — Parallel Sysplex performance: XCF performance considerations

XCF signaling is used to communicate between various members of a sysplex. The user of XCF signaling, usually an MVS component or a subsystem, issue messages to members within the user's group. The content and/or use of these messages are unique to the users of the group.

As XCF messages are generated, they are assigned to a transport class based on group name and/or message size. The messages are copied into a signal buffer from the XCF buffer pool. The messages are sent over outbound paths, (PATHOUT), defined for the appropriate transport class. Messages from other systems are received by inbound paths, (PATHIN). Inbound paths are not directly assigned transport classes, although a correlation can be made about which transport class messages are received via the inbound paths based on the outbound path to which the inbound side is connected.

Transport classes

Transport classes are used to partition messages. Using the CLASSDEF parameter in the COUPLExx parmlib member, you can assign messages to a transport class based on the group name, the message size, or both.

Each transport class has its own resources that consist of a buffer pool and one or more outbound signaling paths. It is recommended you keep the number of transport classes small. In most cases, it is more efficient to pool the resources and define the transport class based on message size. Some initial product documentation recommended separate transport classes for global resource serialization or RMF. These recommendations are no longer advised. If you do have separate transport classes for specific groups based on early product recommendations, you should consider removing those definitions.

With XTCSIZE ENABLED on all systems, avoid using transport classes to partition XCF signal traffic. However, you still need to consider the PATHOUT and MAXMSG resources.

Message Buffers

XCF message buffers are managed by correctly selecting the size of the message most frequently sent from specific buffer pools and by specifying an adequate upper limit for the size of the buffer pool.

Message buffer size

First let's look at the individual message buffer size definitions. Message buffer size is determined by the CLASSLEN parameter on the CLASSDEF statement in the COUPLExx parmlib member. The CLASSLEN value determines the size of the most frequent message expected in this transport class. If a message could be assigned to more than one transport class, XCF selects the one with the smallest buffer that will hold the message. If the signal is larger than the CLASSLEN for any of the assigned transport classes, XCF selects and expands the class with the largest buffer size. Typically, this will be the class with the largest CLASSLEN.

Note: In this section, references to CLASSLEN are a shorthand for "class size". Internally, classes are partitioned by class size, which is derived from the CLASSLEN. The class size is the length of the longest message that fits in the smallest buffer big enough to hold a message of length CLASSLEN. For a CLASSLEN greater than 956, the class size is generally a multiple of 4K less 68. For example, the CLASSLEN values of 1000 and 3000 have the same class size (4028).

Expanding the message buffer entails some overhead. The PATHOUT on the sending side and the PATHIN on the receiving side must be cleared out and expanded to handle the larger buffer size. A new, larger buffer must be obtained on the PATHIN side. If no additional messages of this size are received in a short time period, XCF then contracts the PATHIN, PATHOUT, and buffer sizes. In both of these cases extra XCF internal signals are generated to communicate these changes.

The best way to eliminate the overhead of expanding and contracting the message buffers is to enable the XTCSIZE function switch on all systems in the sysplex. In cases where the XTCSIZE function is not in

use, at least two transport classes are generally needed. One class should be defined with a CLASSLEN of 956 since small messages generally dominate. An additional class or classes would be defined for larger messages.

An example of this specification in the COUPLExx parmlib member is: CLASSDEF CLASS(DEFSMALL) CLASSLEN(956) GROUP(UNDESIG) CLASSDEF CLASS(DEFAULT) CLASSLEN(16316) GROUP(UNDESIG)

The parameter GROUP(UNDESIG) specifies the messages should be assigned to the transport class based solely on message size. This definition makes all the resources available to all users and provides everyone with peak capacity.

There may be times when you want a separate transport class for a specific group. For instance, if you have a particular XCF user which is consuming a disproportionate amount of XCF resources, you may want to isolate this user to a separate transport class to investigate the user's behavior and protect the other XCF users. Hopefully, after you have diagnosed the problem, you can reassign this user to a transport class based on the length of the messages.

You can use an RMF XCF report to determine how well the messages fit. Figure 22 on page 132 shows an example. In this example, the majority of the messages fit in the DEFSMALL class. A few exceeded the size of the DEFAULT class, but not enough to justify the definition of a new transport class.

XCF USAGE BY SYSTEM							

REMOTE SYSTEMS							

OUTBOUND FROM JB0							

TO	TRANSPORT	BUFFER	REQ	----- BUFFER -----			
SYSTEM	CLASS	LENGTH	OUT	%	%	%	%
				SML	FIT	BIG	OVR
JA0	DEFAULT	16,316	189	98	1	1	100
	DEFSMALL	956	55,794	0	100	0	0
JB0	DEFAULT	16,316	176	100	0	0	0
	DEFSMALL	956	44,156	0	100	0	0
JC0	DEFAULT	16,316	176	100	0	0	0
	DEFSMALL	956	34,477	0	100	0	0
TOTAL			134,968				

%SML is the % of messages smaller than the buffer length
 %FIT is the % of messages which fit the buffer length
 %BIG is the % of messages larger than the buffer length

Figure 22. Example: RMF XCF Report

Note: XCF has internal buffers of fixed size: 1K, 4K, 8K, ...64K. XCF uses 68 bytes for internal control blocks. So if you specify a length that doesn't fit one of these sizes, XCF will round up to the next largest size. For example, if you specify 1024, it will not fit into the 1K block (1024-68=956), and XCF will round up to the next largest block. The command, D XCF,CLASSDEF, will list the CLASSLEN specified in the PARMLIB member, in this example, 1024. The RMF XCF report will show the actual buffer length, in this case, 4028.

Message buffer pools

Having determined the optimal size for the individual message buffer, the next thing to do is select an upper limit for the amount of virtual storage to be allocated to the message buffer pool. The message buffer space is virtual storage used by XCF to store the message buffers which are being processed, sent or received.

Most of the virtual storage used for this purpose is backed by fixed central and expanded storage. The storage to hold LOCAL buffers (for communication within the processor) is DREF storage which is backed by central storage. LOCAL buffers are used for messages within groups which are on the same MVS image. Currently APPC and JES3 are the only known IBM exploiters of local messages but OEM applications can choose to take advantage of LOCAL message processing.

XCF only uses the amount of storage it needs; but to insure there are no surprises, the installation can use the MAXMSG parameter to place an upper limit on the amount of storage which can be used for this purpose.

Storage is associated with the transport class, the outgoing paths, and the incoming paths, so MAXMSG can be specified on the CLASSDEF, PATHIN and PATHOUT definitions, or more generally on the COUPLE definition. MAXMSG is specified in 1K units. The default values are determined in the following hierarchy:

OUTBOUND	INBOUND
-----	-----
PATHOUT - not specified, use	PATHIN - not specified, use
CLASSDEF - not specified, use	COUPLE
COUPLE	

The default for MAXMSG is 500 in OS/390 R1 and prior releases. In OS/390 R2 and higher, the MAXMSG default is 750. By not specifying the default parameter, you will automatically get the most current default size as you migrate to newer releases. If you do want a larger value than the default, specify it at the lowest level of the hierarchy as appropriate.

The total amount of storage used by XCF on a single system is the sum of:

- Sum of MAXMSG for all classes multiplied by the number of systems in the sysplex
- Sum of MAXMSG for all PATHOUTs
- Sum of MAXMSG for all PATHINs

Consider the example shown in [Figure 23 on page 133](#).

XCF PATH STATISTICS					
OUTBOUND FROM JC0			INBOUND TO JC0		
-----			-----		
TO	T FROM/TO	TRANSPORT	FROM	T FROM/TO	
SYSTEM	Y DEVICE, OR	CLASS	SYSTEM	Y DEVICE, OR	
JA0	P STRUCTURE		JA0	P STRUCTURE	
	S IXCPLEX_PATH1	DEFAULT		S IXCPLEX_PATH1	
	C C600 TO C614	DEFSMALL		C C600 TO C614	
	C C601 TO C615	DEFSMALL		C C601 TO C615	
	C C602 TO C616	DEFSMALL		C C602 TO C616	
JB0	S IXCPLEX_PATH1	DEFAULT	JB0	S IXCPLEX_PATH1	
	C C600 TO C614	DEFSMALL		C C600 TO C614	
	C C601 TO C615	DEFSMALL		C C601 TO C615	
	C C602 TO C616	DEFSMALL		C C602 TO C616	

Figure 23. Example: XCF Path Statistics

If a MAXMSG of 1000 was specified on the CLASSDEF parameter and MAXMSG was not specified on the other parameters, the maximum storage which could be used by XCF is 22M:

- 2 classes * 3 systems * 1M = 6M
- 8 PATHOUTs * 1M = 8M
- 8 PATHINs * 1M = 8M

Note: This implies if you add additional transport classes, signaling paths or systems, you will be increasing the upper limit on the size of the message buffer pool.

Outbound messages

For the outbound messages to a particular system if the sum of the storage for the CLASSDEF and the PATHOUTs is insufficient, the signal will be **rejected**. This is reported on the RMF XCF report as REQ REJECT for OUTBOUND requests. In general, any non-zero value in this field suggests some further investigation. The problem is generally resolved by increasing MAXMSG on the CLASSDEF or PATHOUT definition. See [Figure 24 on page 134](#) for an example.

XCF USAGE BY SYSTEM						

REMOTE SYSTEMS						

OUTBOUND FROM SYSC						

TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT		ALL PATHS UNAVAIL	REQ REJECT
K004	DEFAULT	956	126,255	...	0	1,391
	DEF16K	16,316	28		0	0
SYSA	DEFAULT	956	97,834		0	0
	DEF16K	16,316	3,467		0	0

TOTAL			227,584			

Figure 24. Example: XCF Usage By System - Outbound Messages

Inbound messages

For the inbound messages from a particular system, if the storage for the PATHINs is insufficient, the signal will be **delayed**. This is reported on the RMF XCF report as REQ REJECT for INBOUND requests. If the delay causes signals to back up on the outbound side, eventually an outbound signal could get rejected for lack of buffer space. In this case, you may wish to increase the MAXMSG on the PATHIN definition. See [Figure 25 on page 134](#) for an example.

XCF USAGE BY SYSTEM						

REMOTE SYSTEMS				LOCAL		
-----				-----		
INBOUND TO SYSC				SYSC		
-----				-----		
.....	FROM SYSTEM	REQ IN	REQ REJECT	TRANSPORT CLASS	REQ REJECT	
	K004	117,613	1,373	DEFAULT	0	
	SYSA	101,490	0	DEF16K	0	

	TOTAL	219,103				

Figure 25. Example: XCF Usage By System - Inbound Messages

Another indicator that the storage for PATHINs is insufficient is the BUFFERS UNAVAIL count on the XCF PATH STATISTICS report. If this is high, check the AVAIL and BUSY counts: AVAIL counts should be high relative to BUSY counts. High BUSY counts can be caused by an insufficient number of paths or a lack of inbound space. First look at the inbound side and see if there are any REQ REJECTs. If so, increase the PATHIN MAXMSG. Otherwise, it is important to review the capacity of the signaling paths. The methodology for determining this is described in [“Capacity planning” on page 137](#).

Note: The RMF Communications Device report cannot be used to determine if the CTC devices are too busy. XCF CTCs will typically always report high device utilization because of the suspend / resume protocol used by XCF.

Local messages

Local messages are signals within the same image, so no signaling paths are required. In this case, the message buffer storage used is the CLASSDEF storage plus any storage specified on the LOCALMSG definition. If MAXMSG is not coded on the LOCALMSG statement the additional message buffer storage contributed is none, or 0 buffers.

Signaling paths

XCF signals from each transport class are sent out on the PATHOUT path and received into the system on the PATHIN paths. Tuning is achieved by altering the number or type of paths, or both. To review the

XCF path configuration use the RMF XCF Path Statistics report. Two different issues commonly reported to IBM regarding signaling paths are reviewed in this flash: no paths defined, and an insufficient number of paths defined.

Number of paths

No Paths: In the worst case, there may be NO operational paths for a transport class. This is not fatal. XCF routes the requests to another transport class but there is additional overhead associated with this operation. To determine if this condition exists, look at the RMF XCF Usage by System report. ALL PATHS UNAVAIL should be low or 0. In many cases, this is caused by an error in the path definition; in other cases, there may be a problem with the physical path. In this example shown in [Figure 26 on page 135](#), the CTC links to system JA0 had been disconnected.

XCF USAGE BY SYSTEM						

REMOTE SYSTEMS						

OUTBOUND FROM SD0						

TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT		ALL PATHS UNAVAIL	REQ REJECT
JA0	DEFAULT	16,316	189	...	0	0
	DEFSMALL	956	55,794		55,794	0
JB0	DEFAULT	16,316	176		0	0
	DEFSMALL	956	44,156		0	0
JC0	DEFAULT	16,316	176		0	0
	DEFSMALL	956	34,477		0	0
TOTAL			134,968			

Figure 26. Example: XCF Usage by System - ALL PATH UNAVAIL

In the **next** example from the same system (see [Figure 27 on page 135](#)), notice for system JA0 there were no paths for the transport class DEFSMALL, so all the requests were re-driven through the DEFAULT class. This caused some queuing (see AVG Q LENGTH of 0.16).

XCF PATH STATISTICS								

OUTBOUND FROM SD0								

TO SYSTEM	T FROM/TO Y DEVICE, OR P STRUCTURE	TRANSPORT CLASS	REQ OUT	AVG Q LENGTH	T FROM/TO AVAIL	BUSY	RETRY	
JA0	S IXCPLEX_PATH1	DEFAULT	56,011	0.16	55,894	117	0	
JB0	S IXCPLEX_PATH1	DEFAULT	176	0.00	176	0	0	
	C C600 TO C614	DEFSMALL	16,314	0.01	16,297	17	0	
	C C601 TO C615	DEFSMALL	15,053	0.01	15,037	16	0	
	C C602 TO C616	DEFSMALL	15,136	0.01	15,136	20	0	
JC0	S IXCPLEX_PATH1	DEFAULT	176	0.00	176	0	0	
	C C600 TO C614	DEFSMALL	11,621	0.01	11,515	106	0	
	C C601 TO C615	DEFSMALL	13,086	0.01	12,962	124	0	
	C C602 TO C616	DEFSMALL	11,626	0.00	11,526	100	0	

Figure 27. Example: XCF Path Statistics - Requests Routed through DEFAULT Class

Is it necessary to correct the 'ALL PATHS UNAVAIL' condition? In most cases, it is. In this example in [Figure 27 on page 135](#), DEFSMALL was defined to hold small messages (956). Because there is no path, they are being re-driven through the **DEFAULT** class. The DEFAULT class is sending data in large buffers (16,316 bytes). This is certainly not an efficient use of message buffer storage to transfer a 956 byte message in a 16,316 byte buffer. Re-driving large messages through a transport class defined with small messages causes more problems. It causes the buffers in this class to expand and contract with all the extra signaling explained previously. Defining separate classes is done for a purpose. If you don't provide paths for these classes, it negates this purpose.

Insufficient Number of Paths: Signaling paths can be CTC links or Coupling Facility structures. In the XCF PATH STATISTICS section example above, the TYP field indicates the connection is a coupling facility structure (S) or a CTC link (C). Since these two types of paths operate in unique ways, different methods are used to evaluate their performance.

- Coupling facility structures

For coupling facility structures, an insufficient number of PATHOUT links could result in an increase in the AVG Q LENG, and BUSY counts high relative to AVAIL counts. Additional paths are obtained by defining more XCF signaling structures in the CFRM policy and making them available for use as PATHOUTs (and/or PATHINs).

Note: RETRY counts should be low relative to REQ OUT for a transport class. A non zero count indicates a message has failed and was resent. This is usually indicative of a hardware problem.

- CTCs

CTCs can be configured in a number of ways. The installation can define CTC's as unidirectional (one PATHOUT or one PATHIN per physical CTC) or bi-directional (one or more PATHOUTs and PATHINs on a physical CTC). Due to the nature of XCF channel programs, a unidirectional path definition can achieve the most efficient use of a CTC thus providing the best XCF response time and message throughput capacity. However, a unidirectional definition will also require using **at least four physical CTCs** to configure for availability. Two paths are sufficient for most systems, thus only those customers with very high XCF activity, (requiring >=4 paths), should consider using the unidirectional definition.

What indicators should be used to determine if there are enough CTCs for a particular transport class? First of all, the AVG Q LEN on the RMF XCF report is **not a good indicator**. In the case of CTCs, queued requests are added to the CCW chain which can increase efficiency. A better indicator to use instead is the Display XCF command. This command was updated by XCF APAR OW38138 to provide the path response time (as seen by XCF). See [Figure 28 on page 136](#) for an example.

D XCF,PI,DEVICE=ALL,STATUS=WORKING							
IXC356I 12.02.12 DISPLAY XCF 901							
LOCAL DEVICE	REMOTE	PATHIN	REMOTE			LAST	MXFER
PATHIN	SYSTEM	STATUS	PATHOUT	RETRY	MAXMSG	RECORD	TIME
C200	JA0	WORKING	C200	10	500	3496	339
C220	JA0	WORKING	C220	10	500	3640	419

Figure 28. Example: Display XCF Command Path Response Time

The MXFER TIME is the mean transfer time in microseconds for up to the last 64 signals received within the last minute. If the MXFER TIME is acceptable, less than 2 milliseconds, (or 2000 microseconds), there is probably enough CTC capacity. To insure capacity for heavier or peak workloads, also check the channel utilization for the CTCs, as reported on an RMF Channel Activity report. In laboratory testing, acceptable XCF message response times were observed even at channel utilization of 70% (or 90% when there were multiple CTCs per transport class). Beyond this threshold, response time degenerated rapidly.

RMF, with APAR OW41317 installed, will store the MXFER TIME as observed in the last minute before the end of the RMF interval in the RMF SMF 74 subtype 2 record.

Type of signaling path

A CTC provides a direct path between two systems, while sending a message through a coupling facility is a two step, push-pull process. Thus, depending on message size and the type of coupling facility link, CTCs are sometimes faster than using coupling facility structures.

These are examples of XCF response time, (MXFER TIME), from controlled experiments in a test environment. The unidirectional CTCs have a single PATHIN or PATHOUT per physical CTC. The bi-directional CTCs have a pair of PATHIN AND PATHOUT defined for physical CTC. The 4 bi-directional CTCs have 4 pairs of PATHIN and PATHOUT per physical CTC.

XCF internally times the various signals and gives preference to the faster paths. For example, in [Figure 29 on page 137](#), compare the number of requests for DEFSSMALL which were sent through the structure to

the number which were sent through the CTCs. It should be noted XCF does not attempt to balance the workload across paths; once it finds a fast path, it continues to use it. APAR OW38138 describes changes which improves the path distribution.

XCF PATH STATISTICS							

OUTBOUND FROM JA0							

TO	T FROM/TO	TRANSPORT	REQ	AVG Q	T FROM/TO		
SYSTEM	Y DEVICE, OR	CLASS	OUT	LNTH	AVAIL	BUSY	RETRY
JC0	P STRUCTURE						
	S IXCPLEX_PATH1	DEFAULT	1,744	0.00	1,176	0	0
	S IXCPLEX_PATH2	DEFSMALL	8,582	0.01	8,362	220	0
	C C600 TO C614	DEFSMALL	20,223	0.01	20,160	63	0
	C C601 TO C615	DEFSMALL	23,248	0.01	23,229	19	0
	C C602 TO C616	DEFSMALL	23,582	0.01	23,568	14	0

Figure 29. Example: XCF Path Statistics

In many environments, the difference in response time between CTCs and coupling facility structures is indiscernible and using coupling facility structures certainly simplifies management of the configuration.

Capacity planning

For availability, a minimum of two physical paths must be provided between any two systems. This can be accomplished with two physical CTCs, structures in each of two different CFs, or a combination of CTCs and coupling facility structures.

Most environments will find the rate of XCF traffic can be handled by the two paths which were configured for availability. Only for environments with very high rates of XCF traffic would additional paths be required.

The XCF message rate capacity of a path is affected by many factors:

1. The size of the message
2. How the paths are defined
3. If the path is also used for other (non-XCF) functions

Based on these factors, message rates (XCF IN+OUT), have been observed from 1000/sec to 5000/sec on a CTC, up to 9000/sec via an ICB and up to 4000/sec per HiPerLink. The adage "Your mileage may vary" is certainly true here.

When using coupling facility structures for XCF messaging, there is also a cost in coupling facility CPU utilization to plan for. As an example, running 1000 XCF messages/sec through an R06 coupling facility would utilize approximately 10% of one coupling facility processor. Additionally, if you use coupling facility structures as XCF paths, make sure the structure size is adequate. You can use the [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062) to obtain an initial estimate for the structure size. If the structure is too small, you will see an increase in the number of REQ REJECT and AVG Q LNTH, and these events will definitely affect response time.

CTC configuration planning

When configuring CTCs for large volumes of XCF traffic some additional configuration planning needs to be done. CTC I/O will use SAP capacity, and large XCF environments can generate I/O rates much higher than traditional DASD and Tape workloads.

The SAP acts as an offload engine for the CPUs. Different processor models have different numbers of SAPs, and a spare 9672 PU can be configured as an additional SAP processor. SAP functions include:

- Execution of ESA/390 I/O operations. The SAP (or SAPs) are part of the I/O subsystem of the CPC and act as Integrated Offload Processor (IOP) engines for the other processors.
- Machine check handling and reset control
- Support functions for Service Call Logical Processor (SCLP)

In high volume XCF environments planning should be done to ensure the CTC configuration is defined so the CTC I/O load is spread across all available SAPs. Information on channel to SAP relationships can be found in the *IOCP User's Guide and ESCON CTC Reference*, GC38-0401-11. Additional Information on 9672 SAP performance and tuning can be found in WSC Flash 9646E at the [IBM Techdocs \(www.ibm.com/support/pages/ibm-techdocs-technical-sales-library\)](http://www.ibm.com/support/pages/ibm-techdocs-technical-sales-library).

Case study

This is a case study that illustrates some of the items discussed.

An application was invoked that was changed to use coupling facility signaling. When the workload was increased XCF delays increased. This was evident from messages like ERB463I which indicated the RMF Sysplex Data Server was not able to communicate with another system because the XCF signaling function was busy.

Looking at RMF Monitor III, it showed the information in [Figure 30 on page 138](#).

RMF 1.3.0 XCF Delays									
Samples: 120		System: J90	Date: 02/07/97	Time: 13.03.00					
Jobname	C	Service	DLY	-----Main Delay Path(s)					
		Class	%	%	Path	%	Path	%	Path
WLM	S	SYSTEM	87	87	-CF-				
MASTER	S	SYSTEM	10	10	-CF-				
RMFGAT	S	SYSTEM	3	3	-CF-				
JESXCF	S	SYSTEM	1	1	C601				

Figure 30. Example: RMF Report - XCF Delays

Comparing the RMF XCF reports to some other reports, it was noticed the amount of XCF traffic had quadrupled and the increase was in the class with the larger CLASSLEN (DEFAULT on this system).

In order to protect other XCF users and to investigate what was happening, a decision was made to separate these messages into their own transport class. A new transport class, NEWXCF, was defined using the GROUP keyword to specifically assign messages from the new application to this class. Since it was known the messages were bigger than the transport class with the smaller CLASSLEN (DEFSMALL), using guess work, it was decided the messages might fit into a 4K(-68) buffer. The report shown in [Figure 31 on page 138](#) was generated.

---- BUFFER ----									
TO	TRANSPORT	BUFFER	REQ	%	%	%	%	ALL	REQ
SYSTEM	CLASS	LENGTH	OUT	SML	FIT	BIG	OVR	PATHS	REJECT
JA0	DEFAULT	20,412	2,167	92	8	<1	100	UNAVAIL	0
	DEFSMALL	956	29,730	0	100	0	0	0	0
	NEWXCF	4,028	106,018	0	0	100	0	0	0
JB0	DEFAULT	20,412	6,132	97	3	<1	100	0	0
	DEFSMALL	956	82,687	0	100	0	0	0	0
	NEWXCF	4,028	18,085	0	0	100	0	0	0

Figure 31. Example: RMF Report - Defining New Transport Class

Since all the NEWXCF messages were too big, the CLASSLEN was increased. [Figure 32 on page 139](#) shows the results.

TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	% SML	---- BUFFER ----			ALL PATHS UNAVAIL	REQ REJECT
					% FIT	% BIG	% OVR		
JA0	DEFAULT	20,412	1,715	90	10	0	0	0	0
	DEFSMALL	956	37,687	0	100	0	0	0	0
	NEWXCF	8,124	103,063	0	100	0	0	0	3,460
JB0	DEFAULT	20,412	2,075	92	8	0	0	0	0
	DEFSMALL	956	38,985	0	100	0	0	0	0
	NEWXCF	8,124	117,727	0	100	0	0	0	195

Figure 32. Example: RMF Report - Increasing CLASSLEN

Now all the messages fit, but some are being REJECTEd. This suggests message buffer space for the outbound path is no longer large enough. As [Figure 33 on page 139](#) shows, the XCF path statistics confirm outbound messages are queuing up.

TO SYSTEM	Y DEVICE, OR P STRUCTURE	TRANSPORT CLASS	REQ OUT	AVG Q LNGTH	AVAIL	BUSY
JA0	S IXCplex_PATH1	DEFAULT	1,715	0.00	1,715	0
	S IXCplex_PATH2	DEFSMALL	486	0.00	486	0
	S IXCplex_PATH3	NEWXCF	103,063	1.42	102,818	245
JB0	C C600 TO C584	DEFSMALL	13,684	0.00	13,644	0
	C C601 TO C585	DEFSMALL	13,603	0.00	13,603	0
	C C602 TO C586	DEFSMALL	12,610	0.00	12,610	0
	S IXCplex_PATH1	DEFAULT	2,075	0.00	2,075	0
	S IXCplex_PATH2	DEFSMALL	737	0.00	737	0
	S IXCplex_PATH3	NEWXCF	117,727	1.26	117,445	282
	C C610 TO C584	DEFSMALL	16,391	0.00	16,391	0
	C C611 TO C585	DEFSMALL	12,131	0.00	12,131	0
	C C612 TO C586	DEFSMALL	12,294	0.00	12,294	0

Figure 33. Example: Insufficient Message Buffer Space for Outbound Path

Increasing the MAXMSG on the PATHOUT for the NEWXCF transport class from 1000 tp 2000 clears up the queuing delays. See [Figure 34 on page 139](#).

TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	% SML	---- BUFFER ----			ALL PATHS UNAVAIL	REQ REJECT
					% FIT	% BIG	% OVR		
JA0	DEFAULT	20,412	2,420	93	7	0	0	0	0
	DEFSMALL	956	41,215	0	100	0	0	0	0
	NEWXCF	8,124	133,289	0	100	0	0	0	0
JB0	DEFAULT	20,412	2,362	93	7	0	0	0	0
	DEFSMALL	956	39,302	0	100	0	0	0	0
	NEWXCF	8,124	143,382	0	100	0	0	0	195

Figure 34. Example: Increasing the MAXMSG on the PATHOUT

The BUSY conditions are reduced, and more importantly the AVG Q LNGTH has been greatly reduced. Since the pathout with the contention is a coupling facility structure AVG Q LNGTH is an appropriate metric to use when tuning. See [Figure 35 on page 140](#).

TO SYSTEM	Y DEVICE, OR P STRUCTURE	TRANSPORT CLASS	REQ OUT	AVG Q LNTH	AVAIL	BUSY
JA0	S IXCPLEX_PATH1	DEFAULT	2,420	0.00	2,420	0
	S IXCPLEX_PATH2	DEFSMALL	361	0.00	361	0
	S IXCPLEX_PATH3	NEWXCF	133,289	0.08	133,117	2
	C C600 TO C584	DEFSMALL	12,700	0.00	12,700	0
	C C601 TO C585	DEFSMALL	16,421	0.00	16,421	0
	C C602 TO C586	DEFSMALL	14,173	0.00	14,173	0
JB0	S IXCPLEX_PATH1	DEFAULT	2,362	0.00	2,362	0
	S IXCPLEX_PATH2	DEFSMALL	1,035	0.00	1,033	2
	S IXCPLEX_PATH3	NEWXCF	143,383	0.09	143,087	296
	C C610 TO C584	DEFSMALL	12,647	0.00	12,646	1
	C C611 TO C585	DEFSMALL	15,944	0.00	15,944	0
	C C612 TO C586	DEFSMALL	12,183	0.00	12,182	1

Figure 35. Example: BUSY Conditions Reduced

When determining how to tune the application to limit the number of XCF messages, a DEF8K transport class for UNDESIG messages was created and the NEWXCF class assigned to this application was eliminated.

Note: In this case study, the messages were being queued because the message buffer space was too small. If, instead of REJECTS, there was a high percentage of messages marked as BUSY, then increasing the number of signaling paths would have been appropriate.

Incidentally the path associated with the NEWXCF was a coupling facility structure that used the new HiPerLinks available on the G3 server. The structure was chosen since it was quicker and easier to implement. Since the structure was receiving over 500 req/sec, it was unclear if the structure could handle the traffic. As can be seen from the queue lengths, it was capable of handling this rate.

Chapter 6. Planning XCF Note Pad Services in a sysplex

An XCF note pad is shared storage that can be accessed by programs throughout the sysplex. Each note pad is hosted in a list structure in a coupling facility, and an additional structure is used to host the note pad catalog. To use the XCF Note Pad Services, the installation needs to define these structures in the Coupling Facility Resource Management (CFRM) policy.

The XCF Note Pad interface supports both authorized and unauthorized callers. The security administrator might have to define System Authorization Facility (SAF) profiles for certain note pads to grant access to the unauthorized callers. For authorized callers, XCF will honor the SAF profile if one is defined. If SAF is not installed, or the installation has not defined a SAF profile for the note pad, XCF rejects the request made by an unauthorized caller and permits the request to go forward for an authorized caller.

In addition, the installation might consider defining SAF profiles for the XCF catalog and note pad structures to prevent anyone but XCF from establishing XES connections to these structures. The data stored in the XCF catalog and note pad structures is managed solely by XCF, and allowing other programs to access the data directly through established XES connections could cause data integrity issues.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on the XCF Note Pad Services.

Steps for setting up XCF Note Pad Services

Complete the following activities to set up XCF Note Pad Services:

- Define the XCF note pad catalog structure in the CFRM policy.
- Define XCF note pad structures in the CFRM policy.
- Authorize XCF note pad requests.
- Restrict IXLCONN access to XCF catalog and note pad structures.

Defining the XCF note pad catalog structure in the CFRM policy

The XCF note pad catalog structure must be defined in the active CFRM policy before a program can use the XCF Note Pad Services.

Chapter 4, “Managing coupling facility resources,” on page 41 provides considerations for setting up a CFRM policy. You format the couple data set that will contain the CFRM policy with the IXCLIDSU utility program. You set up your CFRM policy using the administrative data utility (IXCMIAPU), which resides in SYS1.MIGLIB. See Chapter 12, “Administrative data utility,” on page 335 for more information.

Name of the XCF note pad catalog structure

The name of the note pad catalog structure is SYSXCF_NPCATALOG.

Determining the size of the XCF note pad catalog structure

XCF maintains an entry in the note pad catalog structure for each note pad defined in the sysplex. The size of the note pad catalog structure is proportional to the number of note pads that can be defined in the sysplex. The following table shows sample catalog structure sizes for different note pad counts, assuming the coupling facility is at CFLEVEL 18.

Table 9. Sample catalog structure sizes for different note pad counts		
Number of note pads	INITSIZE (MB)	SIZE (MB)
500	12	14

Table 9. Sample catalog structure sizes for different note pad counts (continued)		
Number of note pads	INITSIZE (MB)	SIZE (MB)
1000	14	17
2000	17	23

The CF Structure Sizer (CFSizer) can also be used to simplify the task of calculating the size of the note pad catalog structure.

The IBM CF Structure Sizer (CFSizer) can be accessed at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062). To size the catalog structure, click on the OEM link and input the catalog structure attributes.

When determining the size of the note pad catalog structure, consider the following required structure attributes:

- A list structure type that has 10 list headers and uses both entry and adjunct data
- A lock table entry count of 0
- Can be altered
- The maximum number of list entries depends on the maximum number of note pads that will be in use in the sysplex at any given time. Use the following formula to calculate the number of maximum list entries:

$$\#max_list_entries = 32 + \#note_pads$$

where *#note_pads* is the maximum number of note pads that will be in use in the sysplex at any given time.

- The maximum number of data elements is ten times the maximum number of entries. For example, if the maximum number of list entries is determined to be 532, then the maximum number of data elements is 5320.
- A maximum of 255 data elements per entry
- A list entry can be located by a key value
- An element increment number of 1 (the size of each element is 256 bytes)
- Both entry keys and secondary keys might be used when processing list entries
- An event monitoring controls (EMC) count of 0
- Resides in a coupling facility of CFLEVEL 9 or higher

After determining the estimated structure size, if you need to change the coupling facility structure size later, you can do so using either of the following two methods:

- Dynamically alter the coupling facility structure size by entering the SETXCF START,ALTER command or issuing the IXLALTER service. However, the actual structure size cannot exceed the maximum structure size value or fall below the minimum structure size value.
- Update the CFRM policy with the new coupling facility size, activate the policy, and then have the operator initiate a rebuild of the structure.

Note: No new note pads can be created in the sysplex if there is not enough space in the note pad catalog structure, and this condition might cause certain applications to fail.

The following references contain more information:

- See [z/OS MVS System Commands](#) for information about the SETXCF command.
- See [z/OS MVS Programming: Sysplex Services Reference](#) for reference information about the IXLALTER service.
- See [z/OS MVS Programming: Sysplex Services Guide](#) for guidance on using the IXLALTER service.

- See [Chapter 4, “Managing coupling facility resources,” on page 41](#) for general considerations for setting up a CFRM policy.
- See [Chapter 12, “Administrative data utility,” on page 335](#) for information about updating the CFRM policy using the IXCMIAPU utility.

Other considerations for the XCF note pad catalog structure

The note pad catalog is used by the XCF Note Pad Services to manage all note pads in the sysplex. If the note pad catalog structure fails, XCF deems all note pads to have failed. Therefore it is important to consider the availability of the note pad catalog structure when defining the structure in the CFRM policy.

XCF does not support user-managed rebuild or user-managed duplexing rebuild for the note pad catalog structure. To increase the availability of the note pad catalog and to avoid a single point of failure, IBM strongly suggests enabling system-managed duplexing for the note pad catalog structure. The note pad catalog structure is only accessed when a major change is made to a note pad (such as note pad creation or deletion), and duplexing the catalog structure does not impact the performance of normal note manipulation requests.

Connectivity should also be considered when defining the coupling facility preference list for the note pad catalog structure. A system must have connectivity to the catalog structure for programs residing on that system to use note pads.

Structure alter is supported for the note pad catalog structure. IBM suggests allowing system-initiated alters (ALLOWAUTOALTER(YES)) for the catalog structure, so the system can automatically manage the space utilization of the coupling facility in which the catalog structure resides.

Deleting the XCF note pad catalog structure



CAUTION: Take care when considering whether to delete the note pad catalog structure. Deleting the note pad catalog structure causes ALL note pads in the sysplex to fail, so you should do so only when no note pads are defined in the sysplex. Use the `DISPLAY XCF,NOTEPAD` command to determine whether there are note pads defined in the sysplex.

If necessary, you can use the `IXCDELNP` utility to delete an XCF note pad. Do not use the utility unless you understand the use of the note pad by its exploiting application and any adverse effects that the application might experience as a result.

The note pad catalog structure is a persistent structure that you can delete with only the `SETXCF FORCE` command or the `IXLFORCE` macro, or it is deleted internally by XCF in a failure scenario. Before you can force the deletion of the structure, you must wait for XCF to disconnect from the note pad catalog structure. If the structure has any connectors, the force command is rejected. There is no command to disconnect XCF. XCF automatically disconnects from the note pad catalog structure if there are no active note pad connectors on the system and no note pad activities (including `DISPLAY XCF,NOTEPAD`) for a time period of approximately fifteen minutes.

Tip: Issue the `D XCF,NP` command, which induces activity, consistently from the same system that already has a connection to the structure. This ensures that activity occurs on only one system and connections are therefore not created or extended unnecessarily.

To ensure that forced structure deletions succeed, ensure that you first understand what causes XCF to establish connections to these structures and when XCF will disconnect from the structures. Then plan the forced deletions for after XCF disconnects. The following steps outline a good procedure for successfully forcing deletions:

1. Issue the `SETXCF FORCE,STR,STRNAME=strname` command. If it succeeds, you are done. If it is rejected because the structure still has connections, continue with these steps.
2. Issue the `DISPLAY XCF,STR,STRNAME=strname` command to determine which systems have connections.
3. Issue the `DISPLAY XCF,NOTEPAD` command to determine which note pads have connections.
4. Shut down the appropriate applications.

5. Try deleting the structures again.

The XCF catalog structure can only be used by the sysplex in which the structure was allocated and formatted. If the sysplex is re-IPled, or if XCF detects that the catalog structure belongs to a different sysplex, it automatically deletes the catalog structure (along with all the note pads) and allocates a new instance of the catalog structure.

The following references contain more information:

- See [Chapter 14, “Deletion utility for XCF note pads,”](#) on page 403 for information about the IXCDELNP utility.
- See [z/OS MVS System Commands](#) for information about the DISPLAY XCF and the SETXCF commands.
- See [z/OS MVS Programming: Sysplex Services Reference](#) for information about the IXLFORCE macro.

Defining XCF note pad structures in the CFRM policy

XCF note pads reside in list structures, and these note pad host structures must be defined in the CFRM policy before an application can start using the XCF Note Pad Services.

XCF looks at the following characteristics when choosing a suitable structure to host a note pad:

- The name of the structure.
- The amount of available storage in the structure.
- Whether there is a list header available to host the note pad. Each note pad structure can host up to 1024 note pads.
- The duplexing attribute of the structure.

Once a suitable host structure is chosen, XCF connects and allocates the structure as needed. If the particular structure is inaccessible (for example, the structure is in rebuild), XCF moves onto the next suitable structure. If there are no other suitable structures, the create note pad request is rejected.

The installation and configuration documentation of the XCF Note Pad exploiter typically provides information that can be used to determine the structure size and attributes needed to host their note pad. You need to define the suitable host structures in the CFRM policy before the exploiter can use the XCF Note Pad Services. Defining multiple host structures allows XCF to use another suitable structure when one is not accessible.

See [z/OS MVS Programming: Sysplex Services Guide](#) for more information about note pad host structure selection processing.

[Chapter 4, “Managing coupling facility resources,”](#) on page 41 provides considerations for setting up a CFRM policy. You format the couple data set that will contain the CFRM policy with the IXCLIDSU utility program. You set up your CFRM policy using the administrative data utility (IXCMIAPU), which resides in SYS1.MIGLIB. See [Chapter 12, “Administrative data utility,”](#) on page 335 for more information.

Naming conventions for the XCF note pad structures

The note pad structure names must match the following conventions:

- IXCNP_ownership, where xx is a two digit hexadecimal number in the range of X'00' to X'FF', and owner is derived from the note pad name specified by the Note Pad Services exploiter when creating a note pad. Owner specific structures are generally defined only if installation wants to isolate a particular set of note pads in a particular set of structures.
- IXCNP_SYSCFxx, where xx is a two digit hexadecimal number in the range of X'00' to X'FF'. These are the names of default or community note pad structures where note pads are to be placed if there are no owner specific structures defined for the note pads.

Note: If you have existing structure names that conform to the naming conventions used by the XCF Note Pad Services, those structure names need to be changed before the XCF Note Pad Services can be used.

Determining the sizes of the XCF note pad structures

XCF maintains an entry in the note pad structure for each note in a note pad. The size of the note pad structure is proportional to the number of notes that can exist in the structure at any given time. The following table shows sample structure sizes for different note counts, assuming that the coupling facility is at CFLEVEL 18:

Table 10. Sample note pad structure sizes for different note counts		
Maximum number of notes “1” on page 145	INITSIZE (MB)	SIZE (MB)
1000	13	18
10,000	21	33
100,000	99	185
500,000	444	858
1,000,000	876	1699
Notes: 1. This is the sum of all notes from all note pads that the note pad structure is expected to host. For example, if the structure is expected to host 50 note pads that each contain 10,000 notes, then the total number of notes that can exist in the structure is 500,000.		

The CF Structure Sizer (CFSizer) can also be used to simplify the task of calculating the sizes of the note pad structures.

The IBM CF Structure Sizer (CFSizer) can be accessed at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062). To size a note pad structure, click on the OEM link and input the structure attributes.

When determining the size of a note pad structure, consider the following required structure attributes:

- A list structure type that has 1040 list headers and uses both entry and adjunct data
- A lock table entry count of 0
- Can be altered
- The maximum number of list entries depends on the maximum number of notes that can exist in the structure at any given time. Use the following formula to calculate the number of maximum list entries:

$$\#max_list_entries = 5736 + \#total_notes$$

where *#total_notes* is the sum of all notes from all note pads that this structure is expected to host.

- A maximum number of data elements that is the same as the maximum number of list entries
- A maximum of 64 data elements per entry
- A list entry can be located by a key value
- An element increment number of 4 (the size of each element is 1024 bytes)
- Both entry keys and secondary keys might be used when processing list entries
- An event monitoring controls (EMC) count of 0
- Resides in a coupling facility of CFLEVEL 9 or higher

After generating the estimated structure sizes from the CFSizer, if you need to change the coupling facility structure size later, there are two ways you can do that:

- Dynamically alter the coupling facility structure size by entering the SETXCF START,ALTER command or issuing the IXLALTER service. However, the actual structure size cannot exceed the maximum structure size value or fall below the minimum structure size value.

- Update the CFRM policy with the new coupling facility size, activate the policy, and then have the operator initiate a rebuild of the structure.

Note: A request to create a new note pad will fail if none of the suitable note pad host structures have sufficient space to satisfy the number of notes that the new note pad needs to support.

The following references contain more information:

- See *z/OS MVS System Commands* for information about the SETXCF command.
- See *z/OS MVS Programming: Sysplex Services Reference* for reference information about the IXLALTER service.
- See *z/OS MVS Programming: Sysplex Services Guide* for guidance on using the IXLALTER service.
- See Chapter 4, “Managing coupling facility resources,” on page 41 for general considerations for setting up a CFRM policy.
- See Chapter 12, “Administrative data utility,” on page 335 for information about updating the CFRM policy using the IXCMIAPU utility.

Other considerations for the XCF note pad structures

Products and subsystems that exploit the XCF Note Pad Services might have preferences as to whether their note pads should reside in a duplexed structure. XCF will try to honor the duplexing preference when creating a new note pad, but will not fail the create request if it is unable to do so. However, hosting the note pads in a less preferred structure might have a negative impact on the performance or availability goals of the exploiter.

Connectivity should also be considered when defining the coupling facility preference list for a note pad structure. A system must have connectivity to the note pad structure for exploiters residing on that system to access the note pads.

Deleting XCF note pad structures



CAUTION: Take care when considering whether to delete a note pad structure. Deleting a note pad structure causes ALL note pads hosted in the structure to fail, so you should do so only when no note pads are defined in the structure. Use the `DISPLAY XCF,NOTEPAD,STRNAME=hostname` command to determine whether there are note pads defined in the structure.

If necessary, you can use the IXCDELNP utility to delete an XCF note pad. Do not use the utility unless you understand the use of the note pad by its exploiting application and any adverse effects that the application might experience as a result.

The note pad structure is a persistent structure that you can delete with only the SETXCF FORCE command or the IXLFORCE macro, or it is deleted internally by XCF in a failure scenario. Before you can force the deletion of the structure, you must wait for XCF to disconnect from the note pad structure. If the structure has any connectors, the force command is rejected. There is no command to disconnect XCF. XCF automatically disconnects from a note pad structure if there are no active note pad connectors or activities that are related to note pads hosted in the structure (including D XCF,NOTEPAD) for a time period of approximately fifteen minutes.

Tip: Issue the `D XCF,NP` command, which induces activity, consistently from the same system that already has a connection to the structure. This ensures that activity occurs on only one system and connections are therefore not created or extended unnecessarily.

To ensure that forced structure deletions succeed, ensure that you first understand what causes XCF to establish connections to these structures and when XCF will disconnect from the structures. Then plan the forced deletions for after XCF disconnects. The following steps outline a good procedure for successfully forcing deletions:

1. Issue the `SETXCF FORCE,STR,STRNAME=strname` command. If it succeeds, you are done. If it is rejected because the structure still has connections, continue with these steps.
2. Issue the `DISPLAY XCF,STR,STRNAME=strname` command to determine which systems have connections.

3. Issue the DISPLAY XCF,NOTEPAD command to determine which note pads have connections.
4. Shut down the appropriate applications.
5. Try deleting the structures again.

The XCF note pad structure can only be used by the sysplex in which the structure was allocated and formatted. If the sysplex is re-IPLed, or if XCF detects that the note pad structure belongs to a different sysplex, it automatically deletes the note pad structure (along with all the note pads hosted by it) and allocates a new instance of the structure.

The following references contain more information:

- See Chapter 14, “Deletion utility for XCF note pads,” on page 403 for information about the IXCDELNP utility.
- See *z/OS MVS System Commands* for information about the DISPLAY XCF and the SETXCF commands.
- See *z/OS MVS Programming: Sysplex Services Reference* for information about the IXLFORCE macro.

Authorizing XCF note pad requests

The XCF Note Pad interface supports both authorized and unauthorized callers. If the calling program runs unauthorized, the installation must define a System Authorization Facility (SAF) profile that grants the program access to the note pad. If the calling program runs authorized, XCF calls SAF to determine whether the program has been granted the access needed to issue the note pad requests. If SAF is not installed, or the installation has not defined a SAF profile for the note pad, XCF rejects the request made by an unauthorized caller and permits the request to go forward for an authorized caller.

If the z/OS Security Server, which includes RACF, or another security product is installed, the security administrator can define profiles that control the use of the XCF note pads. To define appropriate security profiles, the security administrator needs to know the names of the note pads and the types of IXCNONE requests the exploiter wishes to make. For example, a program that needs to create a note pad would need CONTROL access, whereas a program that only reads notes in the note pad needs only READ access. The installation and configuration documentation of the exploiter typically contains information about the names of the note pads (or how to define them) and the types of access required.

The following steps describe how the RACF security administrator can define RACF profiles to control the use of XCF note pads:

1. Define resource profile IXCNONE.owner.application in the FACILITY class.
2. Specify the users who have access to the note pad using the RACF PERMIT command.
3. Make sure the FACILITY class is active, and generic profile checking is in effect. If in-storage profiles are maintained for the FACILITY class, refresh them.

For example, if an installation wants to permit an application with an identifier of SUBSYS1 to create an XCF note pad named NPOWNER1.NPAPP1.NPFUN1.NPQUA1, the security administrator can use the following commands:

```
RDEFINE FACILITY IXCNONE.NPOWNER1.NPAPP1 UACC(NONE)
PERMIT IXCNONE.NPOWNER1.NPAPP1 CLASS(FACILITY) ID(SUBSYS1) ACCESS(CONTROL)
SETROPTS CLASSACT(FACILITY)
```

You can specify RACF user IDs or RACF group IDs on the ID keyword of the PERMIT command. If RACF profiles are not defined, the default allows any authorized user or program (supervisor state and PKM allowing key 0-7) to issue requests for the note pad.

See *z/OS Security Server RACF Security Administrator's Guide* for information about RACF.

Restricting IXLCONN access to XCF catalog and note pad structures

By default, any programs that run in supervisor state or PKM allowing keys 0 to 7 can use the IXLCONN macro to establish XES connections to the XCF catalog and note pad structures. However, the data in these structures is managed solely by XCF, and allowing other programs to access the data directly through established XES connections could cause serious data integrity issues. IBM suggests that installations use System Authorization Facility (SAF) to restrict access to the XCF catalog and note pad structures. No additional action is needed to grant XCF access. Note that restricting access to a note pad structure does not prohibit a program from accessing the note pads hosted in that note pad structure. See [“Authorizing XCF note pad requests” on page 147](#) for more information on defining security profiles to control access to XCF note pads.

The following steps describe how the RACF security administrator can define RACF profiles to control the use of XCF catalog and note pad structures:

1. Define resource profile IXLSTR.structure-name in the FACILITY class. The Universal Access Authority (UACC) should be set to NONE to prohibit access from any programs other than XCF.
2. Make sure the FACILITY class is active, and generic profile checking is in effect. If in-storage profiles are maintained for the FACILITY class, refresh them.

For example, if an installation wants to restrict access to the XCF catalog structure, the security administrator can use the following commands:

```
RDEFINE FACILITY IXLSTR.SYSXCF_NPCATALOG UACC(NONE)
SETROPTS CLASSACT(FACILITY)
```

See [z/OS Security Server RACF Security Administrator's Guide](#) for information about RACF.

Chapter 7. Tuning a sysplex

The performance of systems in a sysplex is closely related to the performance of the signaling service. The throughput and responsiveness of the systems participating in the sysplex depend on the signaling service. Therefore, it is important to be able to identify and correct performance problems that are related to signaling. This information describes considerations for tuning the signaling service and illustrates how you can use RMF reports to do tuning and capacity planning for the signaling service.

It is also important to perform capacity planning so that the signaling service is able to adequately meet the needs of its users, and thereby provide good signaling performance as workloads change over time.

The performance of the coupling facility in the sysplex also affects overall sysplex performance, especially if you are using the coupling facility for signaling.

See [System - Managed CF Structure Duplexing](#) for tuning and performance issues for System-Managed Coupling Facility (CF) Structure Duplexing.

RMF reports and the sysplex

The Resource Measurement Facility (RMF) provides an XCF Activity Report that contains useful measurement data for analyzing the performance of the XCF signaling service and doing capacity planning for an MVS system in a sysplex.

RMF also provides a Coupling Facility Activity Report. It provides information about the usage of a coupling facility, structures within a coupling facility and subchannel connections to a coupling facility in a sysplex.

Additional Information About RMF Report Data: Additional information about the data in the RMF reports can be found in the following publication:

- [z/OS Resource Measurement Facility Report Analysis](#) gives a detailed description of each field on the report as well as tuning suggestions.

XCF activity report

The XCF Activity Report shows the data collected on a system during sysplex processing. Each system collects its own data, and RMF on each system produces reports only about that system's data. It might be necessary to run the RMF reports on two or more systems to get data for corresponding outbound and inbound signaling paths in order to better understand the message traffic patterns.

The XCF Activity Report includes data about:

- Transport classes
- Message buffers and message traffic
- Groups and members
- Signaling paths

Tuning the signaling service

From a tuning standpoint, the primary objective is to ensure that the signal service can maintain throughput and timely message delivery. Typically, an XCF signal path can transfer thousands of signals per second with a few message buffers. In a sysplex where the signal rate between systems is on the order of a thousand signals per second or less, configuring signal paths to avoid single points of failure will likely provide sufficient capacity and resiliency over a wide range of conditions. As the signal rate between systems increases, more care is needed. At higher rates, small delays in signal transfer or spikes in message traffic can quickly lead to large backlogs of signals, which in turn can delay signal delivery until the system works through the accumulated backlog. To maintain timely message delivery,

the configuration must enable the signal service to absorb such spikes and route away from the paths experiencing delay.

Special consideration should be taken in regard to CPU resources and XCF signal performance. Attempting to tune the signaling service without doing so runs the risk of compounding performance issues and causing an outage. For more information, see the [“Importance of Adequate CPU Resources”](#) on page 150.

The remainder of this topic is written under the assumption that the systems in the sysplex have sufficient CPU resources. With that assumption satisfied, the key to ensuring good performance for the signaling service is to provide appropriate amounts of signaling resources, which are signaling paths and message buffers. If the XCF function switch XTCSIZE is not enabled on every system in the sysplex, it might be necessary to configure transport classes as well.

Fundamentally, the objective is to avoid queuing delays that can have a material impact on sysplex performance. Queuing is not necessarily bad or avoidable since it is the system’s way of dealing with temporary disparities between the rate at which signals are produced and the rate at which they are consumed. From a signal performance perspective, the key is to ensure that the queuing delays are short lived. The queuing delays are a function of the queue depth and the rate at which the queued signals can be processed. For inter-system signal traffic, the processing rate is effectively determined by the signal transfer rate. Generally, the signal transfer rate must be taken as a given. It is fundamentally a function of the responsiveness of the underlying hardware (for example, the target system and the CF). So, you don’t really tune the transfer rate. Rather you do your normal hardware configuration and tuning to meet your overall performance objectives, and the signal transfer rate will be what it will be. Historically, the inbound message buffer supply was likely a critical factor as well. However, for systems that support XTCSIZE, the need to manage the inbound buffer supply to maintain timely signal transfer has largely been eliminated.

Thus, regarding minimizing queuing delays, the primary concern is the queue depth. Regardless of the signal transfer time, signal performance is degraded if the queues are deep enough. Queuing can occur for various reasons: the target system is unresponsive, the underlying signal path hardware is unresponsive, a loss of capacity due to signal path failure, increased signal load, the arrival pattern of the send requests, or elongated signal transfer times. The number of signal paths is the primary means of managing the queue depth.

An arbitrarily large number of signal paths cannot be defined since the total potential message buffer usage must be limited to protect the system. Excessive storage consumption by XCF message buffers can degrade system performance and cause system outages. For more information, see [“Number of signal paths”](#) on page 158.

An important protection for the system is to constrain the maximum amount of XCF message buffer space. You can choose to increase the permitted buffer space to resolve “no buffer” conditions, but you must always consider the possible impact to the system if XCF were to consume all that storage. For more information, see [“Tuning the maximum message buffer space”](#) on page 154.

Importance of Adequate CPU Resources

A critical factor for XCF signal performance is responsiveness of the target system. Signal traffic cannot flow at high rates or enjoy short transfer times if the target system is unable to receive and process signals in a timely manner. Therefore, the tuning and diagnosis of signal performance issues must always ensure that the systems in the sysplex have adequate CPU resources. If not, attempts to remedy perceived signal performance issues by increasing the permitted message buffer space or the number of signal paths can compound the performance issues or increase the risk of an outage. For example:

- If a target system lacks the CPU resource that is needed to receive signals, messages might accumulate on the sending systems. Symptoms might eventually include outbound “no buffer” rejects or “all paths busy” conditions. If the target system was responsive, such symptoms might suggest the signal service is being overwhelmed with send requests. Increasing the permitted message buffer space or defining more signal paths might provide relief. However, if the target system is not sufficiently responsive, adding more send side signal resource allows the sending system to accumulate an even larger number of pending signals. This “solution” might expose the sending system to an outage if it permits outbound message buffer usage to exceed levels that are safe for the system. From an exploiter perspective, the

signal delivery time gets longer and longer as more messages are queued. If the target system does become more responsive, the impact of the performance issue persists while the system works through the accumulated backlog of signals.

- If a target system can receive signals but lacks the CPU resource needed to process them, messages might accumulate on the target system. The ability to receive but not process signals is not farfetched. The backend I/O completion processes by which XCF receives signals tend to have priority over other work in the system, including the downstream delivery and processing of those signals by the target members. Symptoms might eventually include inbound "no buffer" or "member stalled" conditions (see IXC431I and IXC638I in *z/OS MVS System Messages, Vol 10 (IXC-IZP)*). If the system had sufficient CPU resources, this symptom might indicate (though not necessarily) that the permitted buffer space should be increased to help maintain signal throughput. However, if the system lacks the CPU resource needed to deliver and process the messages, adding more inbound buffers allows an even larger number of signals to accumulate. This "solution" might expose the receiving system to an outage if it permits inbound message buffer usage to exceed levels that are safe for the system. The larger accumulation of signals awaiting delivery can also compound the lack of CPU resource through long queue effects, making it even more difficult for the system to process messages (or any other work). From an exploiter perspective, the signal delivery time gets longer and longer as more messages are queued. Furthermore, it might elongate the duration of the performance issue by forcing the receiving system to work through an even longer backlog of signals when it does become more responsive.

Since every target system also acts as a sending system, a lack of CPU resources might impact outbound signal traffic. In general, this does not impact XCF signal performance. The number of outbound signals tends to drop if the lack of CPU resources means that the system cannot dispatch the work units that generate them. The reduced signal load means that there is less stress on the signal resources. However, a lack signal service impact does not mean that there is no sysplex impact. The inability to initiate send requests in a timely manner likely means some inter-system, possibly sysplex-wide process, is not completing. The impact of such hangs depends on whose signals are not being sent. The exploiters will likely attribute the hang to poor signal performance since signal delivery appears to be slow. The root cause is the lack of CPU resource. The perceived signal performance problem is not resolved by configuring transport classes, increasing the message buffer supply, or increasing the number of signal paths.

Tuning the message buffer size (class length)

This topic is irrelevant for the XCF defined pseudo-transport class named `_XCFMGD`. The XCF Managed class is used when the XCF function switch `XTCSIZE` is enabled and the target system is running a release of z/OS that supports `XTCSIZE`. XCF determines the `CLASSLEN` specification for `_XCFMGD`. The RMF reports of XCF activity will accurately report data for `SML`, `FIT`, `BIG`, and `OVR` consistent with the definitions of those fields. Since the class size for `_XCFMGD` is 956, `SML` will always be zero. `FIT` and `BIG` show the distribution of message sizes relative to the class size: `FIT` quantifies messages less than or equal to 956 bytes, and `BIG` quantifies messages bigger than 956 bytes. Per traditional transport class reporting, `_XCFMGD` buffers always appear to be tuned for 956-byte messages, so `OVR` will always be 100%.

For a traditional transport class, this information is used to evaluate the suitability of your transport class definitions. However, the inferences made from this information relative to the performance characteristics of a traditional transport class do not apply to `_XCFMGD`. Unlike a traditional transport class, `_XCFMGD` always uses a best-fit buffer when sending a message and an `_XCFMGD` signal path never needs to change its current supported message size when transmitting a message. Thus, in contrast to a traditional transport class where a nonzero `BIG` value typically meant messages bigger than the class size might be incurring additional overhead and delay, there is no such degradation with `_XCFMGD`.

If the XCF function switch `XTCSIZE` is enabled on every system in the sysplex, there is generally no need to define size-only transport classes. Indeed, for optimal XCF signal performance, they should not be defined at all since their existence can skew XCF path selection.

The remainder of this topic assumes you define your own transport classes. However, note that at any point in time, message traffic for a given target system will either be using `_XCFMGD` or your size-only transport class definitions. The counts used to compute `SML`, `FIT`, and `BIG` are not incremented for

your size-only transport classes if _XCFMGD is being used. If you need to assess the suitability of your transport class definitions, you must disable the XTCSIZE switch to get the data needed for that evaluation.

The class length selected for a transport class definition determines the size of the smallest message buffer that XCF provides in a transport class. (Specify class length on the CLASSLEN keyword of the CLASSDEF statement in the COUPLExx parmlib member.)

If a message buffer is too small for a particular message request, the request incurs additional overhead because XCF must format a larger message buffer. If a message buffer is too large for a particular message request, central storage is wasted. XCF adjusts the size of the buffers in a particular transport class on a system by system basis based on the actual message traffic.

The [Figure 36 on page 164](#) provides data that can be useful for evaluating message buffer sizes. This report provides data about the message buffer size for each target system, by transport class. The columns in the report that are relevant to message buffer sizes are:

TO SYSTEM

Indicates the name of the system to which the messages were sent.

TRANSPORT CLASS

Indicates the name of the transport class used to send the messages.

BUFFER LENGTH

Indicates the defined message buffer size used for the transport class. The defined message buffer size, otherwise known as the class size, is the length of the largest message that fits in the smallest buffer big enough to hold a message of length CLASSLEN (the class length specified for the transport class).

REQ OUT

Indicates the number of message requests accepted for delivery to the indicated system in the indicated transport class. The number of message requests accepted for delivery can help you understand the impact of the transport class on the message traffic. If few message requests use a class, consider eliminating the class definition or reducing the resources assigned to the class.

The columns labeled SML, FIT, and BIG under the heading “BUFFER” indicate the distribution of message buffer sizes for messages that were delivered. This distribution is reported in relationship to the defined buffer size, that is, it indicates whether the buffer size needed to accommodate the message is smaller than, equal to, or larger than the defined buffer size.

SML

Indicates the percentage of messages sent that could have used a message buffer smaller than the defined buffer size.

This column is zero if the defined class length causes XCF to use the smallest buffer size that it supports. When this column is not zero, it indicates that some of the messages delivered caused storage to be wasted because the defined buffer size was larger than needed. In this case, consider decreasing the class length or defining a new transport class with a smaller class length. You normally want SML to be close to zero.

FIT

Indicates the percentage of messages sent that fit the defined buffer size. This means that the message would not have fit in the next smaller buffer size supported by XCF, and it would have wasted storage if the next larger buffer size had been used. You want FIT to be as large as possible.

BIG

Indicates the percentage of messages sent that needed a message buffer larger than the defined buffer size. When this column is not zero, it indicates that XCF had to prepare message buffers that were larger than the defined size. Such messages incur additional overhead. XCF may increase the buffer size it provides in order to eliminate this overhead.

If BIG is not zero, consider increasing the class length for the transport class. Or, consider defining a new transport class with a larger class length that can be used to partition the messages by another size. You normally want BIG to be close to zero.

OVR

Indicates the percentage of BIG messages sent that suffered performance degradation. It provides a measure of how successful XCF was in its attempt to satisfy oversized message requests without additional overhead.

Note: Be careful to interpret OVR correctly. It provides the percentage of BIG signals that suffered performance degradation. It is not the percentage of all signals accepted for delivery. For example, if 100 messages were accepted for delivery and one of those messages required a buffer larger than the defined buffer size, then BIG would be 1% and OVR would be 100%. In this situation, OVR being 100% is most likely not significant because only one message had a problem.

Partitioning message traffic

If the XCF function XTCSIZE is enabled on every system in the sysplex, there is no need to partition message traffic by size. XCF automatically maximizes the effective number of outbound message buffers while simultaneously avoiding the overhead and delay associated with changing the message size being used by a signal path. The existence of size-only transport classes skews XCF path selection to favor using not-busy signal paths per the traditional transport class selection rules. This skewing means the best performing signal path might not be used to transfer messages. To achieve optimal signal performance, you should not use transport classes to partition message traffic by size when XTCSIZE is enabled everywhere. The transport class DEFAULT, which is implicitly created by the system, should be the only transport class. In that case, XCF path selection will naturally gravitate towards using the best performing signal paths for all message traffic.

In general, if the XTCSIZE function is not enabled on every system in the sysplex, you may need to partition message traffic according to size to improve signaling performance.

Historically, partitioning signal traffic by size was necessitated by disparities in transfer times for small messages versus large messages. As the technology improved, the differences in transfer time became less significant to sysplex performance. Buffer management and avoidance of the overhead to change signal path buffer sizes became the primary concerns. For systems that support XTCSIZE, the primary concern is efficient use of outbound buffer space and avoiding the overhead to change signal path buffer sizes.

In the [Figure 36 on page 164](#), the SML and BIG columns indicate when it may be useful to partition messages by size. The reports can indicate that such a change might be useful, but not what the appropriate size should be, only that it should be smaller or larger.

Information about message sizes can be determined by creating some temporary transport classes (use the SETXCF operator command) that use different class lengths. Then use the RMF reports to investigate the appropriateness of that class length. By assigning a unique group to those temporary classes, you could determine the range of message sizes used by that group. Alternatively, the message sizes used by the group may have been documented and you could use that information. This information might indicate that a particular group is sending the messages that don't match the defined buffer size. That group can be assigned to its own transport class to isolate it from the other users of the message service.

You can also get information about message size distribution using the DISPLAY XCF command.

- To get information by transport class, use `D XCF, CLASSDEF, CLASS=class` command.
- To get information on sizes for a specific group member, use the `D XCF, GROUP, grpname` command.

Choosing a class length

If you choose a class length equal to the length of the smallest message sent in a transport class, XCF has the maximum flexibility to dynamically make adjustments to larger buffer sizes.

If you choose a class length equal to the length of the largest message sent in the transport class, XCF does not adjust buffer sizes because XCF never provides a message buffer smaller than defined message buffer size.

The smallest message buffer that XCF uses is large enough to accommodate the defined class length. If the class length is equal to the length of the largest message sent in the class, then the smallest buffer

XCF uses can always accommodate the largest message. The advantage is that no message ever needs a larger message buffer and therefore never incurs the extra overhead necessary for XCF to prepare larger buffers.

It is possible that the algorithm used to adjust buffer size does not respond to the particular message traffic pattern at your installation. In this case, you may need to choose a larger class length. The disadvantage of choosing a larger class length is that storage might be wasted by messages that don't need such a large buffer.

When you choose a class length, keep in mind that the class length can impact the amount of message buffer space used for incoming messages (MAXMSG capacity). A large class length implies both larger message buffers and fewer message buffers for a given MAXMSG. Thus, increasing the class length could lead to less capacity to receive on target systems. As a rule of thumb, consider allowing at least 30 message buffers for the PATHIN buffer pool at all times.

Best practices for choosing a class length

The following list contains some recommendations for partitioning message buffers by size and choosing a class length for the classes. You should consider:

- A minimum size class with a class length of 956 bytes.
- A medium size class with a class length of 4028.
- If necessary, define a larger class if in your particular system, you see signals too large to fit in 4K class.

Summary

Use the report shown in [Figure 36 on page 164](#) to evaluate the class lengths chosen for your transport class definitions. SML and BIG can be used to identify mismatches between the defined message buffer size and the size of the messages sent. Use OVR to determine whether a significant number of BIG messages suffered performance degradation. REQ OUT is helpful in determining whether these problems are occurring for a significant number of message requests.

Some of the ways to fix the problems that are highlighted by these numbers are:

- Create one separate class for large messages and assign all groups to that class, by using UNDESIG.
- Define new transport classes to partition message traffic by size.
- Change the class length for the transport class.
- Consider reconfiguring the applications to change the signal distribution.

Note: Enabling the XCF function switch XTCSIZE on every system in the sysplex eliminates the need to create, manage, monitor, or tune size-only transport classes in the manner described above. Instead of defining transport classes, enabling XTCSIZE sysplex-wide is the recommended way to optimize XCF signal performance for different message sizes.

Tuning the maximum message buffer space

Constraining the maximum amount of XCF message buffer space provides an important protection for the system. In general, XCF can maintain signal flow with a few message buffers per path. Additional message buffers are obtained as needed, up to the maximum permitted limits. In some cases, those “as needed” buffers help maintain signal flow. In other cases, the “as needed” buffers allow signals to accumulate on a queue. A “no buffer” condition occurs if XCF is unable to obtain a message buffer due to the maximum permitted limits. The following considerations apply when setting space limits for message buffers:

- Foremost, the buffer space limits must protect the system. However, it can be difficult to determine the amount of buffer space that is safe for the system. It very much depends on the storage demands of the workload and the dynamics of the system. The primary system constraints are the fixed frames below the bar and the virtual storage in the XCF data space. Both areas have a physical limitation of 2 GB. Since both areas are used for purposes other than XCF message buffers, XCF buffer usage must be less than 2 GB. Of the two areas, the below the bar storage is more critical to system performance and

availability. A limit of 800 MB for total inbound buffer space and 800 MB for total outbound buffer space per target system is reasonably conservative. However, since a "safe" value is workload-dependent, some installations might need to be more restrictive while others survive with a larger limit.

- In general, an inbound signal path can maintain signal flow with a small working set of inbound buffers. Additional buffers are then acquired as needed to adapt to conditions, typically increased signal flow or longer delivery times (typically, longer delivery times tend to be the issue). In either case, a buffer is acquired in anticipation of receiving a signal. A concern for the buffer space limit of an inbound path is that a "no buffer" condition impedes timely signal transfer and reduces throughput by delaying receipt of a signal. Historically, "no buffer" conditions might degrade signal performance. Over the years, XCF enhancements have helped mitigate those performance impacts. Today, a system that supports the XTCSIZE function will in general not suffer significant signal performance degradation from inbound "no buffer" conditions, provided an appropriate number of signal paths are configured. Systems that support XTCSIZE provide measurement data to enable installations to assess the "no buffer" impact. The number of "no buffer" conditions is not a reliable indicator of that impact.
- On the send side, a "no buffer" condition implies XCF rejected a request to send a message. These conditions are expected to be short lived. As such, exploiters are encouraged to retry the send request multiple times, perhaps with a short pause between requests. Not all exploiters deal with these rejects gracefully. Although the lack of an outbound buffer does not impact the performance of the XCF signal service, there is potential sysplex impact. From the exploiter perspective, the XCF signal service is unavailable. The inability to send messages in a timely manner likely means some inter-system, possibly sysplex-wide process, is not completing. The impact of such hangs will depend on whose signals are not being sent. So, there is a natural inclination to respond to outbound "no buffer" conditions by increasing the amount of buffer space XCF is allowed to use. However, given the expectation that a few buffers can maintain signal flow, exhausting the outbound buffer supply suggests that signals are being queued. So, either a flood of signals overwhelmed the signal service or signal transfers are taking longer. In essence, increasing the outbound buffer space limit resolves the "no buffer" condition by allowing more signals to be queued. Adding signal paths to increase signal capacity or resolving the signal transfer delays might be a more effective solution for the "no buffer" condition.

Clearly, there are tradeoffs to be made when determining how much storage XCF should be allowed to consume for message buffers. The primary concern must be protecting the system. You can choose to increase the permitted buffer space to resolve "no buffer" conditions, but you must always consider the possible impact to the system if XCF were to consume all that storage. Adding systems to the sysplex or defining additional signal paths automatically increases the amount of storage that XCF is permitted to use. One must always be cognizant of the overall buffer space limits when making such changes.

In general, assume that XCF will at some point consume all the allotted inbound buffer space. This behavior has been observed, particularly in a parallel sysplex running data sharing workloads. It is not unusual for sysplex applications to designate one system as a manager and to then have all systems in the sysplex simultaneously send messages to the manager. Given enough systems and a high enough signal rate, the manager can be overwhelmed and unable to process the messages as fast as they arrive. This then causes inbound message buffers to be queued for delivery to the manager. The backlog of queued signals might consume all the allotted inbound buffer space. XCF manages this situation to ensure that signal traffic continues to flow despite all the buffers in use. Systems that support XTCSIZE can do this more effectively than systems without the support.

Note how critical it is for the system to be sufficiently responsive. A lack of CPU resource might explain why the manager is unable to process its signals in a timely manner. If the system lacks CPU resource, XCF might not be able to adequately mitigate the "no buffer" impacts. In extreme cases, a lack of CPU resource might keep XCF from receiving signals in a timely manner. If so, signals targeted to the unresponsive system might incur queuing delays on the sending systems.

Adjusting message buffer space

Assuming there is not an error condition, you can take the following actions to prevent message requests from being rejected:

1. Increase the message buffer space that XCF is allowed to acquire.

Message buffer space can be increased with the SETXCF command.

To increase the message buffer space for message senders:

- Increase the MAXMSG value for the transport class definition to provide additional default message buffer space for each possible remote system.
- Selectively increase the MAXMSG value for the transport class PATHOUT definition to provide additional default message buffer space for communicating with each possible system.
- Selectively increase the message buffer space for the local system in a particular transport class by increasing the MAXMSG value for LOCALMSG traffic in that class.

To increase the amount of message buffer space available for inbound messages, increase the MAXMSG value for one or more inbound signaling paths (PATHIN) connected to the sending remote system.

Adding signaling paths also increases the amount of message buffer space available.

2. Modify or redistribute the workload so that less message traffic is generated.

One way to reduce message traffic is to change the characteristics of the workload. However, this might not be possible.

The report shown in [Figure 37 on page 165](#) provides a summary of each group's impact on message traffic and can help determine which workloads (that is, groups) would be good candidates to shed or move in order to reduce the message traffic.

3. Tolerate the condition.

You may be able to tolerate the condition if it occurs infrequently or does not degrade performance and if the multisystem applications can tolerate the condition.

Using RMF reports to analyze message buffer space

To determine whether the signaling service was unable to satisfy a message request, the [Figure 36 on page 164](#) provides the following columns:

REQ REJECT

Indicates the number of message buffer requests not satisfied due to message buffer constraints. This column occurs under several headings in the report:

OUTBOUND FROM sysname

Where it indicates the number of requests to send a message to a particular remote system that were rejected in a particular transport class.

INBOUND TO sysname

Where it indicates the number of times that XCF was not able to obtain an inbound message buffer in anticipation of receiving a new message.

LOCAL

Where it indicates the number of requests to send a message within the local system that were rejected in a particular transport class.

In the XCF Activity Report - Path Statistics shown in [Figure 38 on page 166](#), the following columns under the heading "INBOUND TO sysname" are useful:

FROM/TO DEVICE OR STRUCTURE

Which identifies the device number or structure of the inbound/outbound signaling paths that could not obtain message buffers.

FROM SYSTEM

Which identifies the name of the remote system that may have suffered performance degradation because the inbound signaling path could not obtain message buffers.

REQ IN

Which identifies the number of requests.

BUFFERS UNAVAIL

Which indicates the number of times that XCF was not able to obtain an inbound message buffer for the signaling path in anticipation of receiving a new message.

When requests for message buffers are rejected, determine whether the maximum amount of space that XCF is allowed to use for message buffers was too small to support the load (and therefore degrades system performance), or whether there was some unusual/failure condition that caused the limits on buffer space to be reached (in which case the limits performed their intended function because they prevented large amounts of storage from being obtained and further impacting the system).

Analyzing a lack of outbound message buffer space

In the XCF Activity Report shown in [Figure 36 on page 164](#), the “REQ REJECT” columns under “OUTBOUND FROM sysname” and under “LOCAL” indicate the number of times a request to send a message was rejected because XCF could not obtain a message buffer. This count is presented for each remote system, by transport class.

Note that “REQ REJECT” can be incremented more than once for a single message. Some multisystem applications might retry to send a message one or more times before taking some other action (such as setting a timer to retry later). Thus, REQ REJECT reflects the number of attempts to send one or more messages without success.

If outbound messages are being rejected,

- Ensure the target system has sufficient CPU resources. See [“Importance of Adequate CPU Resources” on page 150](#).
- Check the corresponding inbound path at the receiving system. If that system is periodically non-operational, it can cause messages to back up on the outbound path of the sending system. To resolve the problem, you can reduce these outages on the receiving system or increase the message buffer space to allow the messages to be accepted for delivery during these non-operational periods.
- Use the XCF Activity Report shown in [Figure 37 on page 165](#), to see if any of the members in the groups using the transport class have had an unusually high number of send message requests. This might indicate that an error condition is causing the members of that group to generate too many messages. Or perhaps the workload that the group must process simply requires a large number of messages. Resolving the error condition, changing the workload, or providing additional message buffer space are possible solutions.
- Look at RMF reports for the remote system to determine if that system was experiencing inbound message buffer constraints. Such constraints could cause delays in receiving messages and therefore cause outbound messages to back up.
- For local message traffic, message request rejections are most likely due to a runaway sender, delays in receiving messages (such as scheduling delays), or simply not enough space.

Analyzing a lack of inbound message buffer space

In the XCF Activity Report shown in [Figure 36 on page 164](#), the “REQ REJECT” column under the heading “INBOUND TO sysname” indicates the number of times XCF was not able to obtain an inbound message buffer in anticipation of receiving a message over an inbound signaling path. This count is the sum of the number of times this condition occurred for each inbound signaling path and is presented for each possible remote sending system.

The inability to obtain an inbound message buffer does not necessarily indicate a problem. Signaling performance is degraded only if a message is pending on the outbound side of the signaling path. If an inbound message buffer is available before the next outbound message is accepted for delivery, the first message is not delayed. Therefore, use “REQ REJECT” under “INBOUND TO sysname” as an indicator that messages **might** have been delayed.

By correlating the individual inbound signaling path back to the corresponding signaling path on the outbound side, you can determine the transport class to which the outbound path is assigned. If the XCF Activity Report for the outbound system indicates that messages in this transport class targeted to the indicated inbound system were rejected, there likely was an impact.

Even if messages on the outbound path were not rejected outright, there may still have been performance degradation. Use the report the system on the outbound signaling path to determine whether this inability to obtain an inbound buffer delayed the transfer of outbound messages over this signaling path.

Tuning the signaling paths

To achieve the best signaling performance you want a signaling path available to transfer a message when a send message request is made. You should have a path or paths for each class, ideally at least two per class. You might want to provide more paths than appear strictly necessary to handle the average signals per second so that your system can handle bursts of signals. These bursts are hard to predict, in part because they can fall between the normal 15 minute RMF reporting interval.

XCF_TCLASS_CONNECTIVITY in *IBM Health Checker for z/OS User's Guide* will verify that all defined transport classes are assigned at least the indicated number of pathouts (outbound paths).

The RMF reports provide counts of the number of messages sent and received by the system that collected the data. If a system tends to send more messages than it receives, consider providing more outbound signaling paths than inbound paths.

Because the XCF Activity Report provides a breakdown of messages sent in each transport class, you can determine whether a transport class is used to send messages to a particular remote system. If a transport class is not used to send messages to a particular system, signaling paths are not needed in that class for the system. For transport classes that are used, determine if the number of signaling paths provided is sufficient to handle the message traffic. If there are too few paths, signaling performance is degraded; if there are too many, system resources are wasted.

Number of signal paths

Foremost, configure XCF signal paths to avoid single points of failure. Then, as needed, configure additional paths to provide the capacity needed to maintain throughput for the peak average signal load. As needed, configure additional signal paths to address these potential concerns:

- Provide reserve capacity to enable the signal service to maintain throughput should the signal load exceed the expected peak average load. It is a judgment call as to how much white space to allow for such spikes. If they never materialize, you have needlessly configured excess signal capacity. If the system lacks the capacity to deal with such spikes when they do occur, a backlog of pending signals builds. The time a signal spends queued awaiting transfer increases the end-to-end signal delivery time, which might impact sysplex performance. Exploiter send requests might be rejected due to outbound “no buffer” conditions.
- Provide signal paths to manage delays that arise from the signal arrival distribution pattern. For example, if a signal path can transfer 1000 signals/second on average but all 1000 signals are simultaneously queued to the path, the signals at the back of the queue will have a transfer time of one second instead of the expected average transfer time of one millisecond. That worst case delay can be reduced by a factor of N if N signal paths are available instead of just the one. From the perspective of an individual signal path, simultaneous signal arrival means signals are queued to the path while it is busy transferring other signals. The degree to which such queuing is possible is a function of the number of threads that might be sending signals, the average rate at which signals are generated, and the average signal transfer time of the signal path. In practice, the delays attributable to arrival patterns are likely negligible. The number of CPUs available to the system limits the number of threads that might be sending signals. The expected throughput of one signal path is typically an order of magnitude greater than the rate at which signals are generated for a given target system. Thus, such queuing is unlikely to get deep enough to materially impact signal performance. The risk of noticeable impact rises when there is more potential for signals to be queued or remain queued. More CPUs enable more simultaneous threads. Higher signal rates mean more signals arrive more quickly on average. Elongated signal transfer times increase the time signals remain queued for transfer. Overall, installations with systems at distance likely have the greatest risk of being impacted by arrival distribution patterns. Signal transfer time increases with distance. Target systems with inadequate CPU resources also increase the potential for being impacted by arrival distribution patterns.
- Minimize delay by enabling the signal service to route away from signal paths that are temporarily experiencing elongated transfer times. Signal transfer times are highly variable since they are affected by the various dynamics of the system: the recycle time for inbound message buffers, responsiveness of the underlying signal path hardware, and responsiveness of the target system. At times, various conditions can elongate signal transfer times beyond the normal average variation. XCF path selection

intrinsically responds by routing messages to signal paths that are more performant. In effect, the aberrant path appears to be busy and XCF path selection generally tries to avoid busy paths. This strategy is effective when a better performing, not busy signal path is available. Ideally, the signal path configuration is sufficiently rich in diversity and number for there to be unaffected signal paths that can maintain timely signal delivery until the issues affecting the aberrant signal paths are resolved. This strategy will not be effective if there is a common factor that degrades performance of all the signal paths (such as an unresponsive target system).

The number of signal paths cannot be arbitrarily large since there is an associated cost in terms of potential message buffer usage. On the send side, each outbound path increases the maximum amount of outbound message buffer space that might be consumed. On the receive side, every inbound signal path has a maximum amount of inbound message buffer space that might be consumed. Depending on z/OS release and the installed service, these message buffers can consume fixed frames backed above or below the bar (or both) as well as virtual storage in an XCF data space. Inbound buffers also consume common storage. Since excessive storage consumption by XCF message buffers can degrade system performance and even cause system outages, constraints must be imposed on the buffer supply. Thus, the number of possible signal paths must be limited since at some point, the totality of buffer space that might be consumed exceeds values that are safe for the system. Note that 2000 K of message buffer space is generally needed to ensure that any given signal path remains viable.

Signaling path performance

In the XCF Activity Report shown in [Figure 38 on page 166](#), the columns “AVAIL” and “BUSY” provide a breakdown of the number of times the path was selected while available to immediately transfer a message (AVAIL), and the number of times the signaling path was selected while a message was being transferred (BUSY). XCF prefers to use signaling paths that can immediately transfer a message because those paths should provide the least amount of delay in delivering a message. Therefore, interpret the “AVAIL” counts that are high relative to “BUSY” counts as an indication that the signaling service is performing well.

As “BUSY” approaches “AVAIL”, consider adding signaling paths to increase capacity, thereby increasing the possibility that XCF is able to select an idle path for message transfer. Or, consider changing the transport class definitions to change the set of signals making use of the signal paths.

The “BUSY” count of an outbound path can be relatively high if the target system does not have adequate CPU resources. Typically, all systems sending messages to such a system would be similarly impacted.

It is also possible for the “BUSY” count of an outbound path to be relatively high if the inbound signaling path on the remote system does not have sufficient message buffer space to accommodate the inbound message traffic. This can cause signals on the outbound path of the sending system to back up, thereby causing the “BUSY” counts to rise. If the “BUSY” counts are high, run an RMF XCF Activity Report - Path Statistics on the remote system. Look for the column labeled “BUFFER UNAVAIL” under the heading “INBOUND TO sysname”. If the “BUFFER UNAVAIL” count for the inbound signaling path connected to the subject outbound path is not zero, it might indicate that the outbound path's “BUSY” count is high because the inbound path does not have sufficient message buffer capacity to handle the message traffic. Consider increasing the amount of message buffer space that the inbound path can acquire.

Analyzing message counts

In the XCF Activity Report - Path Statistics shown in [Figure 38 on page 166](#), the column labeled “REQ OUT” indicates the total number of attempts made to send a message over the indicated outbound signaling path. Note that summing the “REQ OUT” for all paths leading to a system may not equal the total number of messages sent to that system as presented in the RMF XCF Activity Report - Usage by System, nor will it necessarily equal the sum of the messages sent to that system as presented in the RMF XCF Activity Report - Usage by Member. Often these differences occur because the data for the reports is collected at slightly different times. But it is also the case that there is a subtle difference between the counts in the path statistics report for outbound paths and the counts in the other reports. The path statistics report counts the number of attempts to send a message over a path. The other reports count the number of messages accepted by the signaling service for delivery.

When a signaling path suffers a failure (such as an I/O error), any messages that were not successfully transferred are re-sent on some alternate signaling path. The second attempt to send the message is counted in the path statistics report, but does not show up in the XCF usage reports because the message is accepted for delivery just once. Except for a delay in delivering the message, the re-send of a message is transparent to the message sender. Also, if signaling paths are started and stopped during the monitoring intervals, the path statistics report may not show those devices, hence the messages sent over those paths would not show up in the path statistics report. The XCF usage reports would, however, show those messages.

Impact of signaling path restarts

In the XCF Activity Report - Path Statistics, shown in [Figure 38 on page 166](#), the column labeled “RETRY” counts the number of times the signaling path was restarted and indicates the number of failures the signaling path has had. When this count is nonzero, it indicates that several attempts were made to send one or more messages. A message that cannot be sent on the first attempt suffers performance degradation. Even if all the messages were successfully sent before the path restart occurs (that is, all messages were sent on the first attempt), there may still be degraded performance for the signaling service because a path is not available for message transfer while it is being restarted. Thus, a path restart represents a loss of signaling capacity that may cause the message traffic to be distributed among fewer signaling paths, thereby degrading signaling performance.

Restarts can occur as systems enter and leave the sysplex, or as signaling paths in the sysplex are stopped and started. Barring these kinds of configuration changes, a high restart count suggests that signaling performance suffered because the path was not always operational. XCF automatically stops using a signaling path if path restarts occur so frequently that the device appears to be unusable.

Impact of having no paths in a class

In the XCF Activity Report - Usage by System, which is shown in [Figure 36 on page 164](#), the column labeled “ALL PATHS UNAVAIL” indicates the number of outbound message requests that were migrated to a signaling path in another transport class because there was no operational signaling path that is connected to the intended system and assigned to the selected transport class. A particular transport class might fail to have an operational signaling path for a particular system because:

- There are no signaling paths that are assigned to that class that lead to the other system.
- All the signaling paths that are assigned to that class that lead to the other system have temporarily or permanently failed.

When “ALL PATHS UNAVAIL” is not zero, there is interference between the transport classes. This means that the intended partitioning of message traffic is not obtained. Generally, you want to avoid this condition by assigning signaling paths to the transport class that lead to the intended target system. Messages migrated to an alternative signaling path incur more overhead for XCF to select an alternative transport class. Furthermore, these messages can degrade the performance of messages that are sent in the transport classes that are chosen as alternates; particularly if the buffer size that is used by the alternative class is smaller than the buffer size required by the migrated message.

Thus “ALL PATHS UNAVAIL” can be an indicator of severe problems with the transport class definitions and/or the resources that are assigned to the class. For example, if the transport class did have signaling paths that are defined for the target system but none are working due to a “stall” condition, then it is a possible sympathy sickness indication.

However, the severity of the interference between classes must be considered in light of the number of message requests in the transport class as listed under “REQ OUT”. If these messages occur relatively infrequently, it might not be worth providing a signaling path for their exclusive use.

Tuning the transport classes

In general, transport class definitions are not needed if the XCF function switch XTCSIZE is enabled on every system in the sysplex.

Historically, transport classes were defined to partition signal traffic by size to improve signal performance. In the early days of sysplex, this was necessitated by disparities in transfer times for small messages versus large messages. As the technology improved, the differences in transfer time became less significant. Buffer management and avoidance of the overhead to change signal path buffer sizes became the primary concerns. In general, those concerns are resolved when XTCSIZE is enabled on every system in the sysplex.

You may choose to define transport classes to partition signal traffic by size for certain special cases:

- If any system in the sysplex does not support the XTCSIZE function, all systems in the sysplex likely need to define transport classes. This case typically applies when systems are being upgraded from a release of z/OS without XTCSIZE support to a release of z/OS with XTCSIZE support. In general, you simply continue using the transport class definitions that were in effect prior to the upgrade. Note that all systems in the sysplex use traditional transport class definitions when sending signals to a system that does not support XTCSIZE.
- If the XTCSIZE function switch is enabled, transport class definitions can be used to skew XCF signal path selection when sending signals to systems that support XTCSIZE. This case is typically an intermediate state that applies while systems are being upgraded to a release of z/OS that supports XTCSIZE. For a given message, XCF picks a transport class per the traditional transport class rules and then selects a not-busy signal path from that class to transfer the message. If the class does not have any not-busy paths, XCF can select any signal path connected to the target system. This behavior was intended to help avoid radical changes to signal path usage as the upgraded releases were rolled around the sysplex. Note that selecting a not-busy signal path per the transport class definitions may not provide optimal signal performance. A better performing signal path in a different transport class might provide faster signal transfer. So, in general, it is expected that the transport class definitions will eventually be deleted once XTCSIZE is enabled on every system. However, you might continue to define transport classes if you think the XTCSIZE function switch might be disabled at some point. Otherwise, you would likely need to dynamically reconstruct the transport class definitions before you disable XTCSIZE. Alternatively, you might maintain the class definitions if the skewed path selection served some other purpose. For example, you might want to direct signal traffic towards (or away from) certain coupling facilities. Since small signals are typically the most prevalent, you might assign XCF signal structures to a transport class for small (or big) messages if you wanted the structure to handle more (or less) of the signal traffic.
- If the XTCSIZE function switch is disabled, the system uses transport classes in the traditional manner when sending signals. In this case, you must ensure that the transport class configuration enables the XCF signal service to meet the performance needs of the sysplex. You might choose to disable the XTCSIZE function if it creates issues for your sysplex workload. In general, systems that support XTCSIZE can put more pressure on storage use since inbound message buffer consumption can rise more rapidly in response to signal load. This increased pressure on storage is further compounded when XTCSIZE is enabled on the sending systems since the largest possible buffers are used by the inbound signal path. The increased pressure on storage can lead to outages if XCF buffer space is not limited to values that are safe for the system. Disabling the XTCSIZE function on the sending systems may help reduce the storage use to a level that avoids an outage (ultimately one must limit XCF storage use for message buffers to safe values).

You can also define transport classes to partition signal traffic by group. Typically, such classes are created for one of two reasons: either to provide favored treatment to a group whose traffic is deemed critical, or to isolate an ill-behaved group so it cannot impede the signal traffic of others. An “ill-behaved” group could be one that is under suspicion because it is new or changed. You might isolate such a group to better understand its behavior and impact on the sysplex. In practice, the use of transport classes for these purposes has proven to be problematic or unnecessary. Therefore, the use of transport classes to partition signal traffic based on XCF groups is generally not recommended for production environments. They could potentially be useful for test. If you choose to define such classes, consider the following:

- The signal resources for such classes are available only to the XCF groups assigned to the class, and the groups assigned to the class are restricted to using only the signal resources from the classes to which they are assigned. In effect, the signal traffic of the designated XCF groups is isolated from the signal traffic generated by other groups.

- In a multi-system sysplex, signal paths must be assigned to the transport class to maintain the isolation. If a transport class does not have an operational signal path connected to the target system, its messages will be sent via an operational signal path from some other transport class. That is, messages sent by the designated groups will be intermixed with other message traffic on the signal paths. Their message traffic will not be isolated.
- To ensure good signal performance for the designated groups, assign multiple signal paths to the transport class. So long as there is class connectivity, the signal paths assigned to the class will be used for signal transfer. That remains true no matter how poorly those signal paths perform. Configuring multiple signal paths enables XCF to route message traffic away from a degraded signal path. If that cannot be done, the designated groups may experience elongated signal transfer times and inadequate throughput. Thus, a favored group could be harmed if its transport class is not assigned a sufficiently diverse set of signal paths.
- XCF maintains a working set of buffers based on usage. If the transport class is not used frequently enough, the working set could be empty. Even if the working set of the transport class is nonempty, the working set for one or more of the inbound signal paths on the target systems could be empty if the transport class does not have enough message traffic to keep its signal paths busy. If a message buffer is needed but the working set is empty, XCF may need to allocate a new buffer. Under certain circumstances, the buffer allocation can be delayed. For example, the system might have to wait for fixed frames to be made available to back the buffer. In such cases, the signal transfer for the favored group will be delayed. Note that the working set for classes used by other at large groups (UNDESIG) tend to be nonempty because they generally have more activity. Isolating the favored group to its own transport class denies it access to those readily available buffers.
- The XTCSIZE function does not apply to transport classes that partition signal traffic by group. Therefore, if the designated groups send messages that require different size message buffers, you might need to define multiple transport classes to partition their message traffic by size. This will depend on the nature of the message traffic generated by the designated groups and whether the target systems support XTCSIZE. Assess the need for such classes per the same analysis used to determine the need for size-only transport classes, except that only the message traffic of the designated groups would be considered.

The XCF Activity Report provides data that can be used to determine each group's utilization of signaling resources. This report can identify which groups send and receive the most messages. It is a judgment call as to whether a transport class should be defined to isolate the message traffic for any given group.

Hints

If exploiters of the XCF signaling services have not provided their group name, message size, and message frequency requirements, there are several techniques you can use to determine some of this information. For example, use the XCF Activity Report - Usage by Member to identify group names and the frequency of messages sent and received by each group. Define a transport class and assign to that class just the group under investigation. Use the report to see the distribution of message sizes employed by that group. Adjusting the class length for the transport class definition and examining the report for this class may prove useful to the investigation.

Capacity Planning

For capacity planning, you try to predict how changes in workload will change the requirements for signaling resources. This might be done in order to predict future requirements as additional workload is brought into the system. Or it might be done to predict the impact to signaling resources if the workload is redistributed around the sysplex in a different configuration.

The RMF reports provide counts of the number of messages sent and provide indications of when performance problems occur. For capacity planning purposes, collecting this data over extended periods of time will help you evaluate the impact of changing workloads on the amount of message traffic generated in the sysplex. By periodically running these reports, you will have a history of data with which you can assess the impact of your changing workload. This will be most beneficial if you understand what the workload changes were to be able to associate those workloads with the RMF reports that provide you your signaling data.

Using the RMF reports

The XCF Activity Report - Usage by Member provides counts of the messages sent to and received from each system, by group, member, and system. It also summarizes local messages sent and received, by local group and member.

You can use this report to understand the message traffic loads associated with the various groups and members that communicate with the system. With this understanding, you can do capacity planning. A group is most likely to be either directly responsible for a workload, or to be performing a service function on behalf of a variety of workloads. Even if you are not able to associate a particular workload with a particular group, you can still use the reports to collect data that describes the history of your installation's use of the signaling resources. This data can be used to predict future needs. If a workload can be directly tied to a particular group, it may be possible to understand the impacts to signaling resources when that workload is moved to some other set of systems in the sysplex.

Remote Message Traffic

In the XCF Activity Report - Usage by Member under the heading labeled "MEMBERS COMMUNICATING WITH sysname", are counts of messages sent to and received from remote systems in the sysplex. The name of the group and member and the remote system on which the member resides are identified, respectively, under the columns GROUP, MEMBER, and SYSTEM. The column "REQ FROM sysname" provides the count of messages sent by one or more members on system "sysname" (the system on which the data was collected) to the indicated member (that resides on the remote system). The column "REQ TO sysname" provides the count of messages sent to one or more members on system "sysname" (the system on which the data was collected) by the indicated member (that resides on the remote system).

Use your historical data to predict what signaling resources will be needed as changing workloads impact the remote signaling performed by these groups. For example, you might predict that additional signaling paths will be required at some point in the future to provide sufficient signaling capacity. Or you might determine that fewer signaling paths would be needed if the members of the remote system were moved to the local system or if some workload that uses services provided by the group were moved to the local system. You might not always be able to move a group member because the multisystem application may not have been structured to permit this kind of placement of members.

You can use the XCF Activity Report - Path Statistics to watch the "BUSY" and "AVAIL" counts on a signaling path. If "BUSY" grows relative to "AVAIL" over time, it suggests that the message traffic workload is increasing. You might want to add additional signaling paths to maintain signaling performance.

In the XCF Activity Report - Usage by Member under the heading labeled "MEMBERS ON sysname", are counts of messages sent and received by each member on system "sysname", the local system that collected the data. The name of the group and member is identified respectively under the columns GROUP and MEMBER. The column "REQ OUT" indicates the number of messages sent by the member. The column "REQ IN" indicates the number of message received by the member. Use this data to determine which members on the local system are responsible for most of the message traffic.

If there is only one member per group on each system, you might be able to determine the precise message traffic patterns between particular group members. However, this is not possible to do in the general case.

Local message traffic

In this report, the "REQ OUT" count includes all messages sent by the indicated member, including any messages sent to members (possibly itself) on the local system. Similarly, "REQ IN" includes all messages received by the indicated member, including those sent by members (possibly itself) on the local system.

To determine how many messages were sent within the local system by a particular group, take "REQ OUT" and "REQ IN" and discard those signals that were sent to or received from another system. For example, to determine how many signals were sent by members of a local group, take the total for the "REQ OUT" values for all the members of that group as reported under the heading "MEMBERS

ON sysname”. Then take the total “REQ FROM sysname” value for that group from under the heading “MEMBERS COMMUNICATING WITH sysname”. Subtract the total “REQ FROM” value from the total “REQ OUT” value and you will have the total number of signals sent locally by that group. With this data you can predict what signaling resources will be needed to support local message traffic in the future.

Using the DISPLAY command

You can supplement your information about message traffic from the RMF reports using information returned from the DISPLAY command. Issue the following command to show the message count for each buffer size in the transport class:

```
DISPLAY XCF,CLASSDEF,CLASS=classname
```

RMF XCF Activity Report – Sample

This topic includes a sample XCF Activity Report. For a description of RMF reports, see *z/OS Resource Measurement Facility User's Guide*.

Also, see *z/OS MVS Programming: Sysplex Services Guide* and *z/OS MVS Programming: Sysplex Services Reference* for descriptions of the IXCMG and IXLMG macros, which can also be used to obtain tuning and capacity planning information.

[Figure 36 on page 164](#) is an example of an XCF activity report, showing usage by system.

X C F A C T I V I T Y													
PAGE 1		MVS/ESA SP5.1.0		SYSTEM ID SYS1 RPT VERSION 5.1.0				DATE 03/14/94 TIME 16.30.00		INTERVAL 15.00.000 CYCLE 1.000 SECONDS			
XCF USAGE BY SYSTEM													

REMOTE SYSTEMS												LOCAL	

OUTBOUND FROM SYS1										INBOUND TO SYS1			SYS1

TO REQ SYSTEM REJECT	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	----- % SML	BUFFER % FIT	----- % BIG	ALL % OVR	PATHS UNAVAIL	REQ REJECT	FROM SYSTEM	REQ IN	REQ REJECT	TRANSPORT CLASS
SYS3 DEFAULT	DEFAULT	0	956	334,956	0	100	0	0	0	SYS3	335,259	0	
TOTAL			334,956							TOTAL		335,259	

Figure 36. XCF Activity Report - Usage by System

[Figure 37 on page 165](#) is an example of an XCF activity report, showing usage by member.

X C F A C T I V I T Y									
PAGE 2		MVS/ESA SP5.1.0		SYSTEM ID SYS1 RPT VERSION 5.1.0		DATE 03/14/94 TIME 16.30.00		INTERVAL 15.00.000 CYCLE 1.000 SECONDS	

XCF USAGE BY MEMBER									

MEMBERS COMMUNICATING WITH SYS1					MEMBERS ON SYS1				
-----					-----				
-----					-----				
GROUP	MEMBER	SYSTEM	REQ FROM SYS1	REQ TO SYS1	GROUP	MEMBER	REQ OUT	REQ IN	
COFVLFNO	SYS3	SYS3	52	3	COFVLFNO	SYS1	52	3	
TOTAL			52	3	TOTAL		52	3	
					SYSATB01	M222 M223	0 0	0 0	
					TOTAL		0	0	
SYSATB02	M95 M96	SYS3 SYS3	0 0	0 0					
TOTAL			0	0					
SYSDAE	SYS3	SYS3	0	0	SYSDAE	SYS1	0	0	
TOTAL			0	0	TOTAL		0	0	
SYSGRS	SYS3	SYS3	335,177	335,177	SYSGRS	SYS1	335,177	335,177	
TOTAL			335,177	335,177	TOTAL		335,177	335,177	
SYSMCS	SYS3	SYS3	39	65	SYSMCS	SYS1	39	65	
TOTAL			39	65	TOTAL		39	65	
SYSMCS2	SYS3	SYS3	0	0	SYSMCS2	SYS1	0	0	
TOTAL			0	0	TOTAL		0	0	
SYSWLM	SYS3	SYS3	14	14	SYSWLM	SYS1	14	14	
TOTAL			14	14	TOTAL		14	14	

Figure 37. XCF Activity Report - Usage by Member

Figure 38 on page 166 is an example of an XCF activity report, showing path statistics.

X C F A C T I V I T Y												
PAGE 3	MVS/ESA SP5.1.0			SYSTEM ID SYS1 RPT VERSION 5.1.0			DATE 03/14/94 TIME 16.30.00			INTERVAL 15.00.000 CYCLE 1.000 SECONDS		
TOTAL SAMPLES = 900				XCF PATH STATISTICS								

OUTBOUND FROM SYS1								INBOUND TO SYS1				

TO	T FROM/TO		TRANSPORT	REQ	AVG Q				FROM	T FROM/TO		REQ
Y	DEVICE, OR									Y	DEVICE, OR	
BUFFERS	P	STRUCTURE	CLASS	OUT	LNTH	AVAIL	BUSY	RETRY	SYSTEM	P	STRUCTURE	IN
SYSTEM												
UNAVAIL												
SYS3	C 0CA0	TO 0CA0	DEFAULT	91,864	0.04	90,789	1,075	0	SYS3	C 0CA1	TO 0CA1	
84,966	0											
	C 0C80	TO 0C80	DEFAULT	86,081	0.02	84,990	1,091	0		C 0C81	TO 0C81	
78,088	0											
	C 0EA0	TO 0EA0	DEFAULT	87,490	0.03	86,423	1,067	0		C 0EA1	TO 0EA1	
82,012	0											
	C 0E80	TO 0E80	DEFAULT	77,021	0.02	75,955	1,066	0		C 0E81	TO 0E81	
94,293	0											
TOTAL				-----					TOTAL		-----	
				342,456								339,359

Figure 38. XCF Activity Report - Path Statistics

Tuning UNIX System Services performance in a sysplex

You can use shared file systems in your sysplex for use by UNIX System Services. The use of a shared file system may affect response time and throughput on file systems being shared in a sysplex because of the increase in XCF message traffic. Therefore, you should monitor XCF signalling reports closely, in order to get the benefit that shared file systems offer, such as additional availability and recoverability.

For example, SY1, a client system, requests a read on a file system mounted on SY2. Using shared file system support, SY1 sends a message requesting this read to SY2 via an XCF messaging function:

```
SY1 ==> (XCF messaging function) ==> SY2
```

After SY2 gets this message, and the file becomes available for a read, SY2 gathers the requested data from the file. If the file is frequently accessed, SY2 may be able to simply retrieve the data from its cache. SY2 then returns the data via the same route the request message took:

```
SY2 ==> (XCF messaging function) ==> SY1
```

The increase in message traffic requires that you use the appropriate RMF reports to monitor both message buffer and signaling path utilization. It may be necessary to increase the number of message buffers and to define additional XCF signaling paths.

Tuning coupling facilities

The Coupling Facility (CF) provides hardware assistance to the management of global resources and workloads in a Parallel Sysplex environment. Requests to the CF can be synchronous (the sending processor waits for the completion of the request) or asynchronous (the sending processor continues with other work until it recognizes that the request has completed). Sometimes, when SYNC requests encounter delays, they are changed to ASYNC requests.

Several key indicators can be used to determine whether the coupling facility is performing well:

- “Coupling facility activity report” on page 167
- “SYNC and ASYNC service” on page 171
- “CHNGD requests” on page 172

- [“Potential sources of change and delay in service times” on page 172](#)

See also the *PR/SM Planning Guide* for tuning information.

Coupling facility activity report

The coupling facility activity report provides summary information about the use of each coupling facility in the sysplex, detailed information about each active structure in the coupling facility, and summary information about subchannels connecting the coupling facility to the systems in the sysplex.

How z/OS performs synchronous to asynchronous conversion of coupling facility requests

z/OS V1R2 introduced a new heuristic for determining whether it is more efficient in terms of the z/OS host (sender) system's CPU utilization, to issue a synchronous command to the coupling facility synchronously or asynchronously. With this support, referred to simply as “the heuristic” throughout the remainder of this topic, z/OS will automatically and dynamically control the issuing of CF requests either synchronously or asynchronously, as needed, so as to optimize CPU consumption related to CF requests on the sender z/OS system.

As background, z/OS has the ability to issue requests to the coupling facility either:

- Synchronously, which means that the processor that issued the request waits or spins until the operation is complete at the coupling facility
- Asynchronously, which means that the processor that issued the request is freed up to run another task rather than synchronously waiting or spinning, while the command is transferred to and executed at the coupling facility

In general, asynchronous CF operations take somewhat longer to complete than synchronous operations - their elapsed time or service time is longer than that of a similar request that was issued synchronously. In addition, they require more software processing to complete the request once the CF has completed the asynchronous execution of the command. Asynchronous requests also tend to impact the efficiency of the underlying host processor hardware due to context switching between tasks, scheduling of units of work to process the back-end completion of the asynchronous CF request, suspending and resuming the unit of work that initiated the CF request, and other overhead associated with the asynchronous processing.

However, despite all the additional costs associated with asynchronous request processing, from a total CPU capacity impact standpoint, a sufficiently long-running synchronous CF operation will eventually cost more than the same operation processed in an asynchronous fashion, simply due to the opportunity cost of the processor having to wait or spin for a long time while the CF operation is occurring. The faster the z/OS processor issuing the CF request is, the greater this opportunity cost appears to be, for a given CF service time, because a faster processor might have done more work than a slower processor during the time spent waiting for the CF request to complete. And of course, the longer the CF operation is likely to take, the greater this opportunity costs becomes, in terms of work that the processor might have been able to execute during the execution of the CF request.

Therefore, as the processor executing the CF request gets faster, or as the CF operations themselves take longer (for any reason), it becomes less attractive to perform CF operations synchronously, and more attractive to perform those operations asynchronously. At some threshold, a crossover point is reached where issuing the operation asynchronously costs less in terms of sender CPU overhead than issuing that same operation synchronously would have cost.

In releases prior to z/OS 1.2, z/OS used a relatively simple and static algorithm to try to manage this tradeoff between the efficiency of synchronous and asynchronous CF operations. In those releases, lock structure requests were never converted to asynchronous execution, and list or cache requests were converted from synchronous to asynchronous execution based on some simple rules of thumb, including:

- Certain intrinsically long-running types of CF commands, for example, commands that scan through large sets of entries within the structure, were always converted to asynchronous.

- CF commands involving large (for example, >4K bytes) data transfers to or from the CF were always converted to asynchronous.
- Particular combinations of processor types for the sending z/OS system and the receiving CF were recognized and singled out for additional opportunities for conversion to asynchronous.

While these simple rules of thumb somewhat accomplished their intended purpose, they did not handle some very important conditions which were specific to certain configurations or workloads, for example:

- GDPS, in which a CF might be located a considerable distance from the sending z/OS processor. A particular CF request might complete in a short time if executed on a nearby CF, but would take much longer if sent to a distant CF at the other GDPS site, due to speed-of-light latencies. In this case, it could be more efficient to convert the requests to the distant CF to asynchronous execution, but still drive similar CF requests to the local CF synchronously.
- Highly variable workloads, in which fluctuations in the CF workload (and hence, the CF processor utilization) cause fluctuations in the synchronous CF service time, independent of the inherent processor speed of the CF.
- Low-weighted CF partition with shared CPUs, which tends to greatly elongate the synchronous CF service times, independent of the inherent processor speed of the CF. A CF with shared processors will tend to yield good service times during those intervals when the CF is actually dispatched on a shared processor, but will tend to yield very poor service times during intervals where the CF is NOT dispatched on a shared processor, and thus it cannot possibly service the request.

In z/OS V1R2 and higher, a much more sophisticated heuristic was provided to explicitly recognize the crossover point between the cost of synchronous and asynchronous CF request processing, in a general manner for all combinations of processor configurations, CF link technologies, types of workloads, ranges of CF utilizations and other variations in CF responsiveness, such as distance-related latencies for example.

The heuristic drives requests to the CF in the appropriate synchronous or asynchronous execution mode, based on the actual observed service times for similar requests. At the same time, CF lock structure requests were also enabled for asynchronous execution based on the heuristic, similar to list and cache structure requests.

The heuristic dynamically monitors actual observed synchronous CF request service times, at a fine granularity, for each type of CF command, on a per-CF basis (and on a per-pair-of-CFs basis for those CFs that can participate in System-Managed CF Structure Duplexing). This monitoring also takes into account the amount of data transfer requested on the operation, and several other request-specific operands that can significantly influence the service time for the request. These observations are then recorded in a table, organized by request type and the other factors described above, in which z/OS maintains a moving weighted average synchronous service time for each specific category of request. This moving weighted average is biased towards recent history, so that z/OS can react responsively to factors that may suddenly affect a CF's performance, such as a sudden workload spike.

The heuristic then compares these actual observed synchronous service times against dynamically calculated thresholds, in order to determine which kinds of operations would be more efficient if they were to be executed asynchronously. z/OS calculates several different thresholds for conversion – reflecting the fact that the relative costs of asynchronous processing for list, lock, and cache requests are not the same, nor are the relative costs of asynchronous processing for simplex and duplexed requests the same. All of the calculated thresholds are then normalized based on the effective processor speed of the sending processor (which in turn incorporates information about both the intrinsic processor speed of the machine, and multiprocessing effects based on the number of CPs that are online for the z/OS image at the time), to accurately reflect the opportunity cost of synchronous execution for the processor on which z/OS is executing.

As CF requests are submitted for processing, the characteristics of each request are examined to determine the recently-observed performance of similar requests. If z/OS determines that similar requests have been experiencing good performance, in other words, that the moving weighted average service time of those requests is below the calculated threshold for conversion, then z/OS will direct the current request to be executed synchronously. If z/OS determines that similar requests have been experiencing poor performance, that is the moving weighted average service time of those requests is

above the calculated threshold for conversion, then z/OS will convert the current request to asynchronous execution if it is possible to do so.

Note that z/OS will occasionally sample synchronous performance for CF requests even when the heuristic has determined that it would be more efficient to perform the request asynchronously, by performing every Nth request of a given type synchronously. This mechanism ensures that if some factor should change which results in the CF's performance improving, that improvement will be observed and acted on. Thus, the heuristic does not convert 100% of the requests to asynchronous execution; a small percentage of requests are always issued synchronously.

The following are some other considerations to be aware of, relative to the z/OS heuristic:

- CF exploiters may explicitly request asynchronous execution of their requests. Such requests are not subject to heuristic conversion at all, z/OS simply drives them asynchronously as requested. These requests are NOT reported as CHNGD on RMF reports, and are reported as ASYNC requests.
- CF requests may be converted to asynchronous execution due to the lack of subchannel resources that are needed to process them. Such requests are converted to asynchronous and queued for later execution when a subchannel becomes available. The heuristic is not involved in this kind of conversion. These requests are reported as CHNGD on RMF reports, and are reported as ASYNC requests.
- CF requests may be converted to asynchronous execution due to serialized list or lock contention. The heuristic is not involved in such conversion. These requests are NOT reported as CHNGD on RMF reports. They undergo various forms of software processing related to contention resolution, and may in fact perform several individual CF accesses as that contention management processing is performed, and those CF accesses may be performed in a CPU-synchronous fashion even though the request itself has become asynchronous with respect to the requestor.
- CF requests that are converted to asynchronous execution due to the heuristic are NOT reported as CHNGD on RMF reports, and they are reported as ASYNC requests.
- As a result of the way CHNGD requests are counted and reported, a high percentage of CHNGD requests is indicative of a lack of subchannel resources and a corresponding lack of sufficient CF link connectivity to the CF in question. The CHNGD counter does not reflect anything related to conversion for other reasons, including the heuristic.
- When averaged over time and for a mixture of request types, heuristic conversion is seldom all-or-nothing. One generally sees a mix of synchronous and asynchronous requests over an interval of time; when the synchronous service time is high you will tend to see more asynchronous requests, and when the synchronous service time is low you will tend to see more synchronous requests. Generally, there are always a small percentage of requests that remain synchronous, due to service time sampling.
- In a configuration where some CFs are local and some CFs are remote with respect to a given z/OS system, there will tend to be more conversion to asynchronous execution for requests going to the remote CF (because of the higher service time caused by distance latency). The farther away the CF is located, the more likely conversion becomes.
- When something in the configuration changes such that more heuristic conversion takes place, it is normal to observe an increase in total CPU consumption for the workload. As synchronous service times get longer, CPU consumption inevitably increases, because each synchronous CF access takes longer and therefore causes the processor to wait/spin for a longer time. When service times get long enough, however, the heuristic will tend to cap this growth in CPU consumption by converting to asynchronous execution. Once that threshold is reached, further increases in synchronous service time will **not** result in further increases in CPU consumption, since the CPU cost of performing a CF operation asynchronously is fixed, and largely independent of the service time.

So, when a configuration change (for example, the introduction of distance latencies into a sysplex that was previously located in one physical site) suddenly results in significant amounts of heuristic conversion to asynchronous execution, in general one sees an increase in CPU consumption on the sending z/OS systems – but that increase will be less than it otherwise would have been, had the conversion to asynchronous execution not taken place.

Figure 39 on page 170 shows that an increase in CF service time from A to B, crossing the service time threshold for conversion, will definitely result in an increase in CPU consumption (from C1 to C2) even

with the heuristic. However, that increase is less than it would have been (C1 to C3) had the heuristic not capped the CPU utilization increase:

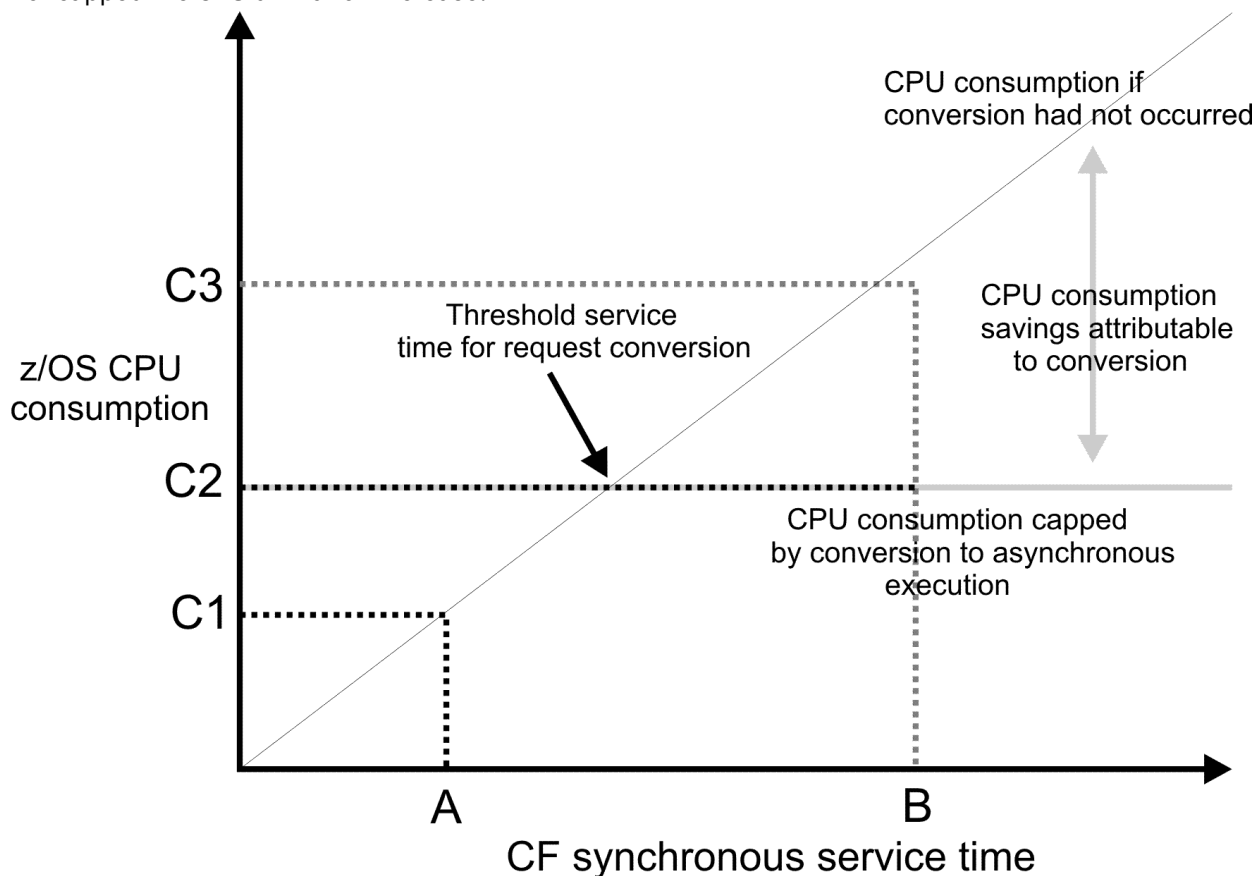


Figure 39. CF synchronous service time

- Upgrades to faster z/OS processors may suddenly result in significant increases in the amount of heuristic conversion, since the calculated thresholds for conversion will be lower, the faster the sender processor is. The purpose of the heuristic is to automatically and dynamically react to such configuration changes, and adjust the flow of requests accordingly to so as to optimize z/OS CPU consumption.
- More generally, any configuration or workload changes that affect the observed CF request service times, or result in a change to the calculation of the conversion threshold, may result in substantial changes in the amount of heuristic request conversion that is observed. Examples of such changes include:
 - Changing to a faster or slower type of CF link technology.
 - Upgrading or downgrading a z/OS processor to a faster or slower technology, or changing the number of online processors in a z/OS image (which changes the effective processor speed somewhat).
 - Upgrading or downgrading a CF processor to a faster or slower technology.
 - Adding or removing CF processors from a CF image.
 - Changing the shared/dedicated nature of the CF processors, or, for a CF with shared processors, changing the LPAR weight of the CF partition, or enabling/disabling the use of Dynamic CF Dispatching.
 - Changes in the amount of distance latency that occurs between the sending z/OS and the target CF.
 - Changes in the CF Level of the target CF.
 - Changes in CF utilization, caused by workload changes or other factors (such as placement of additional structures in a particular CF).
- z/OS CPU time accounting is affected by the heuristic conversion of CF requests. When a CF request is processed synchronously, generally all of the CPU time spent processing the request is attributed to the

address space that initiated the CF request. However, when some or all of these requests are processed asynchronously, the CPU time will get split among the address space that initiated the CF request, and the XCFAS address space. This is because a significant portion of the back-end processing for an asynchronous CF request occurs in the XCF address space. Minor components of the CPU consumption may occur in other address spaces as well.

You can use either of the following methods to modify the thresholds that are used for synchronous to asynchronous conversion:

- The SYNCASYNC statement of the COUPLExx parmlib member
- The **SETXCF MODIFY, SYNCASYNC** command

Using either method, you can independently vary the thresholds that apply to the following classes of requests:

- Simplex non-locking requests
- Duplex non-locking requests
- Simplex locking requests
- Duplex locking requests

Reducing a threshold makes it more likely that a request of the corresponding type will process asynchronously; increasing a threshold makes it more likely that the request will complete synchronously. For example, if you increase a threshold to try to achieve a greater throughput rate, you must accept a greater CPU cost to do so. IBM recommends that if you choose to modify the thresholds, you should keep the values close to the system determined default values. Setting the values much lower or much higher than the default thresholds can significantly affect the performance of coupling facility requests. You can determine the default values from the output of the **DISPLAY XCF, COUPLE** command.

To summarize, at z/OS 1.2 and above, the heuristic for synchronous to asynchronous conversion of CF requests is capable of dynamically recognizing, in a very general and responsive way, situations in which z/OS host processor capacity/efficiency is being impacted by poor CF request synchronous service times, and taking the appropriate action by converting those requests to asynchronous execution, thereby capping that impact. The resulting higher elapsed time for the requests that are converted to asynchronous execution should, in general, have no noticeable effect on end-user response time for a given transaction processing workload.

SYNC and ASYNC service

You can use the number of SYNC requests that were changed to ASYNC requests (CHNGD requests) as an indicator of coupling facility performance. The SYNC and ASYNC service times, as well as the CHNGD requests, are reported on the RMF Coupling Facility Structure Activity report, as seen in [Figure 40 on page 171](#).

COUPLING FACILITY ACTIVITY														PAGE	1	
OS/390 REL. 01.03.00		SYSPLEX UTCPLXJ8 RPT VERSION 1.3.0				DATE 05/01/1997 TIME 16.00.00		INTERVAL 030.00.000 CYCLE 01.000 SECONDS								

COUPLING FACILITY NAME = CF1																

COUPLING FACILITY STRUCTURE ACTIVITY																

STRUCTURE NAME = COUPLE_CKPT1		TYPE = LIST				-----										
SYSTEM NAME	# REQ TOTAL	-----		REQUESTS			-----									
	AVG/SEC	# REQ	% OF ALL	SERV AVG	TIME(MIC) - STD_DEV	REASON	# REQ	% OF REQ	---- /DEL	AVG TIME(MIC) - STD_DEV	----- /ALL	EXTERNAL CONTENTIONS				
J90	9114	SYNC	3127	3.4%	296.6	118.3	NO SCH	1831	30.6%	997.4	2362	305.0	REQ TOTAL	897		
	5.06	ASYNC	5949	6.5%	1524.9	2108.8										
		CHNGD	38	0.0%	INCLUDED IN ASYNC											

Figure 40. Example: RMF CF Structure Report

In this example, during a 30 minute interval, the JES2 Checkpoint structure had 3127 SYNC requests that completed in an average service time of 296.6 microseconds. There were 5949 ASYNC requests

and 38 SYNC requests that were changed (CHNGD) to ASYNC requests. The ASYNC and CHNGD requests completed in an average of 1524.9 microseconds.

CHNGD requests

There are two varieties of SYNC requests: immediate SYNC requests that must be completed as soon as possible and non-immediate SYNC requests that can be delayed, if necessary. If MVS determines that a non-immediate SYNC request will be delayed because of resource problems (for example, the subchannels are unavailable), it will change the request to an ASYNC request. These requests are counted in the CHNGD field of the RMF CF Structure Activity report and their service times are included in the ASYNC service times. However, requests that are converted from SYNC to ASYNC only for heuristic reasons based on service time considerations, are not counted as CHNGD. Therefore, CHNGD can be used as an indicator of whether or not there are sufficient CF links or subchannels available to accommodate the workload.

As a rule of thumb, the total number of requests CHNGD should be less than 10% of the total requests to a given structure.

For more information, see [“How z/OS performs synchronous to asynchronous conversion of coupling facility requests”](#) on page 167.

Potential sources of change and delay in service times

If the service times or the percentage of requests changed exceed the recommended values, there are many potential sources of delay you can take a look at. In general, any configuration or workload changes that affect the observed CF request service times, or result in a change to the calculation of the conversion threshold, may result in substantial changes in service time. Examples of such changes are listed as follows. In subsequent sections under this topic, the indicators of some problems and corrective options are discussed.

- Insufficient CF capacity
- Contention for CF paths
- IOP contention
- Shortage of CF subchannels
- Significant increase of CF subchannels.
- Changing to a faster or slower type of CF link technology.
- Upgrading or downgrading a z/OS processor to a faster or slower technology, or changing the number of online processors in a z/OS image (which changes the effective processor speed somewhat).
- Upgrading or downgrading a CF processor to a faster or slower technology.
- Adding or removing CF processors from a CF image.
- Changing the shared/dedicated nature of the CF processors, or, for a CF with shared processors, changing the LPAR weight of the CF partition, or enabling/disabling the use of Dynamic CF Dispatching.
- Changes in the amount of distance latency that occurs between the sending z/OS and the target CF.
- Changes in the CF Level of the target CF.
- Changes in CF utilization, caused by workload changes or other factors (such as placement of additional structures in a particular CF).

Insufficient CF capacity

If both the ASYNC and SYNC service times for the various structures exceed the recommended values, the first thing to check is the CF utilization. The RMF CF Usage Summary Reports shows the AVERAGE CF UTILIZATION of the processors in the CF; see [Figure 41 on page 173](#) for an example.

COUPLING FACILITY ACTIVITY											PAGE 1
OS/390 REL. 01.03.00		SYSPLEX UTCPLXJ8 RPT VERSION 1.3.0			DATE 05/01/1997 TIME 16.00.00		INTERVAL 030.00.000 CYCLE 01.000 SECONDS				

COUPLING FACILITY NAME = CF1											

COUPLING FACILITY USAGE SUMMARY											

STRUCTURE SUMMARY											

TYPE	STRUCTURE NAME	STATUS CHG	ALLOC SIZE	% OF CF STORAGE	# REQ	% OF ALL REQ	AVG REQ/ SEC	LST/DIR ENTRIES TOT/CUR	DATA ELEMENTS TOT/CUR	LOCK ENTRIES TOT/CUR	RECLAIMS
LIST	IXCPLEX_PATH2	ACTIVE	31M	1.5%	996232	16.3%	553.46	7611	7596	N/A	
	LG2_DFHLOG_001	ACTIVE	26M	1.3%	44159	0.7%	24.53	61 21K 15K	86 85K 42K	N/A N/A N/A	
LIST	COUPLE_CKPT1	ACTIVE	13M	0.6%	71760	1.2%	39.87	3192 3007	3182 3007	2 1	0
LOCK	IRLMLOCK1	ACTIVE	31M	1.5%	1627K	26.7%	904.00	114K 27	0 0	4194K 384	
CACHE	RLSCACHE01	ACTIVE	244M	12.1%	714874	11.7%	397.15	323K	92K	N/A	
	VSAMCACHE1	ACTIVE	34M	1.7%	657723	10.8%	365.40	323K 183K 101K	92K 0 0	N/A N/A N/A	

PROCESSOR SUMMARY											

AVERAGE CF UTILIZATION (% BUSY)			17.3	LOGICAL PROCESSORS:			DEFINED 6	EFFECTIVE 6.0			

Figure 41. Example: RMF CF Usage Report

Generally, best performance from the CF is obtained when the average CF utilization is less than 50%. If the utilization exceeds this, there are various actions that can be taken, depending on the environment.

- Verify that the number of logical processors defined is correct. There have been instances when the CF was configured to one logical processor during service and was not reset.
- Verify that the processor resource allocated to the CF is what you expected. If this is an LPAR environment and the CF partition is sharing CPs, the processor resource will be limited by the definition of the WEIGHT and CAP parameters and contention from the other partitions. If this CF resides on a processor with MVS images, information about the WEIGHT and CAP definitions can be found on the CPU Partition Data Report. The amount of processor resource available to the CF is reported as LOGICAL PROCESSORS EFFECTIVE. You will get the best response time by dedicating CP(s) to the CF partition.
- If you have multiple stand-alone CFs in your configurations, check the distribution of structures. If one CF has much higher utilization than the other, redistribute the structures based on ALLOC SIZE and # REQ.
- If all else fails, you can increase the capacity of the CF by adding CPs or obtaining an additional CF.

Contention for CF paths

When a CF request obtains a subchannel, in most cases, it will proceed down the path to the CF and complete the processing with no further delays. However, if this is a PR/SM environment with multiple z/OS images sharing coupling facility links, the request can encounter a "path busy" condition.

A path busy condition occurs when a coupling facility operation is attempted on a subchannel and at that moment physical resources ("link buffers") that are associated with any of the currently online CHPIDs for the target coupling facility are not available. Without an available link buffer to use, the system cannot start the request. The request is rejected synchronously with a path busy indication that z/OS handles by re-driving the request until a link buffer is available; at that time, the request is successfully accepted for delivery to the coupling facility.

Path busy conditions generally occur only when coupling CHPIDs are shared across multiple z/OS LPARs and when overcommitment of the number of active subchannels exists across the sharing LPARs relative to the number of link buffers. Each coupling CHPID provides a fixed number of physical link buffers that

can be used to process coupling facility requests. Each sharing z/OS image is assigned a number of subchannels equal to the number of link buffers; therefore, each sharing z/OS image is capable of using the entire link buffer capacity that is associated with the shared CHPID. However, if multiple sharing z/OS images try to initiate a large enough number of coupling facility requests, all of the available link buffers in the shared CHPID become saturated before the aggregated subchannels across all of the sharing LPARs are saturated. This excess demand causes path busy conditions to occur. In effect, you can consider path busy conditions as the contention caused by too many subchannels chasing after too few link buffers.

Path busy conditions are handled by z/OS synchronously re-driving them. The amount of time spent doing such re-drive processing is included in the reported service time for the coupling facility request. Additionally, z/OS counts and reports the total number of coupling facility requests that experienced path busy conditions before being successfully initiated. Therefore, there are two ways to observe the occurrence and effects of path busy conditions:

- By determining the percentage of requests that experienced one or more path busy conditions (count of requests that experienced one or more path busy conditions divided by the total number of coupling facility requests)
- By observing an elongation of the service time associated with coupling facility requests, representing the time spent re-driving those requests.

Note that many different sysplex configuration and workload-related factors can affect the occurrence of path busy conditions, such as the following:

- Insufficient total link capacity (that is, too few shared CHPIDs, given the coupling facility workload)
- "Over-shared" link capacity (that is, CHPIDs shared across too many z/OS LPARs, given the coupling facility workload)
- Intensive coupling facility workloads that drive too many concurrent coupling facility requests (enough to saturate all available link buffers) across all sharing LPARs during steady-state operation.
- Workloads that normally do not drive enough concurrent coupling facility requests to saturate the available link buffers except during occasional "bursts" when the link buffers can become saturated.
- Workloads that contain many long-running coupling facility operations, which occupy link buffers for relatively long periods of time.
- Various configuration factors that cause increased coupling facility service times, such as extended link distance (for example, in GDPS environments), use of system-managed duplexing, high coupling facility utilization, shared-processor coupling facility images, and so forth. These conditions all cause coupling facility operations to occupy link buffers for relatively long periods of time.

The system reports the total number of requests that experienced one or more "Path Busy" conditions on the RMF CF Subchannel Activity report in the column labeled PTH BUSY.

Evaluating Path Busy Conditions and taking Actions to Reduce Them

When evaluating the occurrence of path busy conditions, remedial action is generally not required unless the percentage of requests that experienced one or more busy conditions (count of count of requests that experienced one or more path busy conditions divided by the total number of coupling facility requests) is greater than 10%.

You can reduce the impact of excessive path busy conditions through one or more of the following:

- Add more shared coupling CHPIDs that connect to the coupling facility to increase the total number of link buffers that are available to the sharing z/OS LPARs.
- Decrease the "extent of sharing" of the coupling CHPIDs. Consider sharing the same number of coupling CHPIDs across fewer z/OS LPARs, or even dedicating coupling CHPIDs to just one z/OS LPAR.
- Attempt to eliminate one or more of the configuration or workload conditions described above that might affect the occurrence of path busy conditions.

z/OS Subchannel Tuning

z/OS supports a function called subchannel tuning to dynamically manage and tune the number of in-use subchannels across z/OS LPARs that are sharing coupling CHPIDs. z/OS attempts to reduce the overcommitment of the number of subchannels relative to the underlying shared physical link buffers that can cause path busy conditions. To perform subchannel tuning, each z/OS image monitors its own path busy percentage with respect to each coupling facility over short monitoring time intervals. At the end of each of these intervals:

- If the observed path busy percentage is "high," z/OS reduces the number of in-use subchannels in the partition by a small amount while keeping a minimum number of subchannels available for use at all times.
- If the observed path busy percentage is "low," z/OS adds back some of the subchannels that were previously removed from use.
- Otherwise, z/OS takes no action to change the number of subchannels currently in use.

z/OS makes these changes to gradually tune the number of subchannels in use within each sharing partition and attempt to achieve an acceptable level of path busy as seen by that partition. By reducing the number of subchannels available for use, z/OS can convert the "spin" to re-drive path busy conditions into a software subchannel queueing model (as requests queue up to wait for the scarcer subchannels to become available).

You can see the dynamic effects of subchannel tuning in RMF reports or reports of similar monitoring products. At the end of each monitoring interval the RMF or equivalent report shows the number of configured subchannels and the current "snapshot" of the number of subchannels that are actually in-use. The number of in-use subchannels can fluctuate from interval to interval as a result of subchannel tuning actions. Because the time granularity of z/OS subchannel tuning is very fine (measured in seconds) compared to RMF monitoring intervals (typically 30 or 60 minutes), subchannel tuning actions might have occurred many times during the RMF interval. The reported path busy rates, subchannel queueing rates and other measurements reflect the overall net effects given whatever subchannel tuning occurs during the interval.

IOP contention

If the ASYNC service times exceed the guidelines, one possible cause could be a delay in the IOP microcode. The IOP (or SAP) handles I/O to DASD, CTC traffic and ASYNC CF requests prior to HiPerLinks. Any one of these could encounter a delay in the IOP, which in turn would delay the others.

To determine if this delay is occurring, look at the RMF I/O Queuing Activity report. The AVG Q LENGTH records only I/O and CTC requests, but a queue length greater than 1.0 suggests delays are occurring and should be investigated. In Figure 42 on page 175, for example, the large AVG Q LENGTH of 7.58 was due to a Director Port busy condition.

I/O Q U E U I N G A C T I V I T Y							
MVS/ESA SP5.1.0		SYSTEM ID J80					
TOTAL SAMPLES = 7200		RPT VERSION 5.1.0					
		IOP	ACTIVITY RATE	AVG Q LENGTH			
		00	406.534	7.58			
LCU	CONTENTION RATE	DELAY Q LENGTH	% ALL CH PATH BUSY	CHAN PATHS	CHPID TAKEN	% DP BUSY	% CU BUSY
000C	4.176	0.02	0.06	29	1.895	73.33	0.32
				35	1.870	73.75	0.15
000F	65.181	23.10	0.06	29	12.203	67.32	2.49
				35	12.240	60.11	3.87

Figure 42. Example: RMF CF Subchannel Activity Report

Shortage of CF subchannels

Another possible cause for higher service times and excessive queuing is a shortage in the number of CF links from the sending CECs. Modern link types support up to 32 subchannels per CHPID (with 1x InfiniBand). The data to be sent to the CF is loaded into these subchannels. If no subchannels are available, ASYNC requests are queued. Non-immediate SYNC requests are changed to ASYNC requests (which are then also queued). Immediate SYNC requests *spin* waiting for the next available subchannel in most cases; while some locking requests can be changed to ASYNCH requests and then be queued.

As of z/OS V1R10, some locking operations and IXLSYNCH operations can support a queuing protocol when no subchannels are available, which is similar to the current handling of CF list and cache requests. This enhancement is made to eliminate the costly CPU-synchronous spins, and thus reduce the significant CPU overhead when processors must wait for an active CF operation to complete so that its subchannel can be re-used to drive the next request.

Data about the delayed requests is reported on the RMF CF Subchannel Activity report as SYNC and ASYNC DELAYED REQUESTS and summarized on the TOTAL line. For those requests which experience a delay, the duration of the delay is reported separately (/DEL). (Some of these fields were added or changed by RMF APAR OW13536 which was included in OS/390 R2. WSC FLASH 9609 describes the changes and provides a detailed information about the data on the reports).

You can assess the impact of the subchannel delay, by looking at the /ALL field. This takes the delay time for the requests that were delayed and amortizes it over all the requests. So, for example, consider a system whose 1930 SYNC requests encountered a subchannel busy. Each of these is delayed for an average of 461.2 microseconds. Because the guideline for SYNC request service times is 250-350 microseconds, this looks like a problem. However, when this delay is amortized over all the SYNC requests under /ALL, the report indicates only 3.8 microseconds.

As a guideline, the sum of the SYNC and ASYNC requests delayed (TOTAL - % OF REQ) should be less than 10% of all requests. If the percentage of requests queued is greater than this, you should:

- Verify the configuration is what you expected. This can be done by looking at the RMF CF Subchannel activity report. The number of subchannels configured (SCH GEN) tells you how many subchannels were defined in the IOCDs. (HCD automatically defines 2 subchannels for each CF link). RMF also reports the number of subchannels which are currently being used (SCH USE) and how many CF links are currently connected (PTH). If the number of subchannels currently being used are less than the number of subchannels defined, verify that you have not lost connectivity on some CF links.
- If all the CF links are connected, you may want to consider additional CF links or upgrading to a faster CF link technology, or both. If the CF is totally configured for CF paths, you may want to consider moving a structure to another CF facility or adding another CF.

Usage of coupling facilities by structure

For coupling facilities at CFLEVEL=15 or higher, you can now get information on coupling facility usage for both the entire processor and broken down by structure. (At a pre-CFLEVEL=15 coupling facility, the coupling facility activity report just shows coupling facility usage as a whole.)

Note that not all processor time is directly accountable to a particular structure, so that the usage for the structures will not add up 100% of the processor busy time.

Chapter 8. Planning sysplex availability and recovery

When work on one system in a sysplex cannot complete because the system fails, other systems in the sysplex remain available to recover the work and continue processing the workload. The goals of failure management in a sysplex are to minimize the impact that a failing system might have on the sysplex workload so that work can continue, and to do this with little or no operator intervention. You should be aware that in some cases sysplex delays may occur while other systems attempt to recover work from a failing system. See [“Lock structure considerations” on page 211](#) for examples of sysplex delays that may occur.

Note: When a remaining system in the sysplex is running on a z15™ server that supports recovery process boosts, the system will request System Recovery Boost's recovery process boost support to expedite the recovery of work from the failed system and continue processing the workload. The following processes will trigger a system to request System Recovery Boost to accelerate recovery processing:

- XCF group services system-wide recovery cleanup for members on removed systems
- Coupling facility datasharing member recovery for lock structure connectors
- CFRM LOSSCONN recovery management
- MVS-initiated structure rebuild due to loss of connectivity and coupling facility failure
- Structure duplexing failover and re-establishing of structure duplexing after loss of connectivity, structure failure or coupling facility failure.

For more information on the use of System Recovery Boost by sysplex services, see [z/OS MVS Programming: Sysplex Services Guide](#).

For more information on the use of System Recovery Boost by sysplex services, see: IBM Z System Recovery Boost Content Solution (www.ibm.com/support/z-content-solutions/system-recovery-boost/) or IBM Documentation (www.ibm.com/docs/en/zos).

The actions MVS is to take in failure situations is determined by the information specified through the COUPLExx parmlib member, the SFM policy, the XCFPOLxx parmlib member, the automatic restart management policy, the use of the system status detection (SSD) partitioning protocol, and system defaults.

• COUPLExx Parmlib Member

From COUPLExx, MVS obtains basic failure-related information, such as when to consider a system to have failed and when to notify the operator of the failure. (COUPLExx parmlib specifications might be different for each system, depending on its workload, processor capacity, or other factors.)

• SFM Policy

If all systems in a sysplex are running OS/390 or MVS/ESA SP Version 5, you can use the sysplex failure management (SFM) policy to define how MVS is to handle system failures, signaling connectivity failures, or PR/SM reconfiguration actions. Although you can use SFM in a sysplex without a coupling facility, to take advantage of the full range of failure management capabilities that SFM offers, a coupling facility must be configured in the sysplex.

SFM makes use of some information specified in COUPLExx and includes all the function available through XCFPOLxx.

The SFM policy also can be used in conjunction with the REBUILDPERCENT specification in the CFRM policy to determine whether MVS should initiate a structure rebuild when loss of connectivity to a coupling facility occurs.

• XCFPOLxx Parmlib Member

In a multisystem sysplex on a processor with the PR/SM feature, XCFPOLxx functions can provide some of the same capabilities as those provided by the SFM policy. XCFPOLxx functions are also referred to as the XCF PR/SM policy.

• Automatic Restart Management Policy

Use the automatic restart management policy to specify how batch jobs and started tasks that are registered as elements of automatic restart management should be restarted. The policy can specify different actions to be taken when a system fails, and when an element fails. Automatic restart management uses the IXC_WORK_RESTART exit, the IXC_ELEM_RESTART exit, the event exit, and the IXCARM macro parameters, in conjunction with the automatic restart management policy (the specified values and the defaults) when determining how to restart elements.

• System Status Detection (SSD) Partitioning Protocol Using BCPii

XCF uses the SSD partitioning protocol and BCPii services to enhance and expedite sysplex partitioning processing of systems in the sysplex. With BCPii services, XCF can automatically detect when a system in the sysplex has become demised. Then XCF can initiate partitioning the demised system immediately, bypassing the failure detection interval and the cleanup interval and avoiding the need for system fencing and manual operator intervention. A system image is considered demised when XCF determines that the system is removable from the sysplex without further delay. The system might encounter one of the following conditions:

- The system enters a non-restartable disabled wait state.
- The system experiences a LOAD operation.
- The system has experienced a RESET or other equivalent action (such as system reset, checkstop, and power-down).

• System Default Status Update Missing (SUM) Action

If no active SFM policy or PR/SM policy is defined, the default SUM action is used in response to a status update missing condition. Before z/OS V1R11, the default SUM action is to prompt the operator when a system is detected to be status update missing. As of z/OS V1R11, if a system is in the status update missing condition and is not sending any XCF signals, the system is to be isolated immediately using the fencing services through the coupling facility.

Other sysplex and couple data set failures, such as those caused by a power failure, might require operator intervention. See [“Handling concurrent system and couple data set failures”](#) on page 197.

Controlling availability and recovery through COUPLExx

The following topics describe the recovery-related parameters in COUPLExx.

Planning the failure detection interval and operator notification interval

Planning the failure detection interval: On each system in the sysplex, MVS periodically updates its own status and monitors the status of other systems in the sysplex. A *status update missing condition* occurs when a system in the sysplex does not update its status information within a certain time interval. This time interval is the failure detection interval. The effective failure detection interval is the larger one of the user-specified failure detection interval and spin failure detection interval:

- The user-specified failure detection interval comes from the following sources:
 - The INTERVAL keyword in COUPLExx (explicitly or by default)
 - The SETXCF COUPLE,INTERVAL command
 - The value set by cluster management instrumentation software

When the value is omitted, the default INTERVAL value equals the spin failure detection interval.

- The spin failure detection interval is derived from the excessive spin parameters specified in the EXSPATxx parmlib member. The value is computed as follows, where N is the number of excessive spin recovery actions, +1 indicates the implicit SPIN action, and SpinTime is the excessive spin loop timeout interval:

```
spinfdi = (N+1)*SpinTime + 5
```


When EXSPATxx default specifications are used, the default spin failure detection interval can be 45 seconds or 165 seconds depending on the configuration. See *z/OS MVS Initialization and Tuning Reference* for more details.

Exception: You can enable the USERINTERVAL option on the COUPLExx parmlib member or on the SETXCF FUNCTIONS command to force the user-specified INTERVAL value to be the effective failure detection value, even though it is smaller than the spin failure detection interval.

Planning the operator notification interval: The operator notification interval is the amount of time between when a system no longer updates its status and when another system issues message IXC402D to inform the operator of the condition. You specify the operator notification interval on the OPNOTIFY keyword in COUPLExx..

After IPL, you can modify the failure detection and operator notification intervals using the SETXCF COUPLE command.

The OPNOTIFY value can be specified as an absolute value or as a relative value.

- When an absolute value is specified, the effective OPNOTIFY value used by the system is the greater one of the specified OPNOTIFY value and the effective failure detection interval.
- When a relative value is specified, the effective OPNOTIFY value used by the system is the sum of the effective failure detection interval plus the specified relative OPNOTIFY value.

Considerations: The operator notification interval must be equal to or greater than the failure detection interval. For many installations, the default values for the failure detection interval and the operator notification interval are reasonable.

If you are not using the default failure detection interval, you need to evaluate the tradeoff between removing a failing system immediately to prevent disruption to your sysplex and tolerating a period of non-productivity so that the system has a chance to recover.

If you specify these intervals, consider the following:

- If SFM is active in the sysplex, and ISOLATETIME is specified, then the operator is not prompted. However, if SFM fails to isolate the failing system, the operator is still prompted.

If SFM or XCFPOLxx is active and RESETTIME or DEACTTIME is specified, then the operator notification time runs in parallel to the RESETTIME or the DEACTTIME.

Otherwise, to ensure that the operator receives timely notification of a failure, specify an operator notification interval that is the same as or only slightly greater than the failure detection interval.

- XCF signaling uses the failure detection interval of the inbound system to determine when an outbound signaling path is inoperative. If signals are queued for transfer longer than the receiving system's failure detection interval, the path is restarted.

Planning the cleanup interval

When MVS is removing a system from a sysplex, the cleanup interval is the maximum number of seconds that MVS waits for members of a group to clean up their processing before putting the system on which they reside in a wait state. (Note that this is not a required protocol — some groups, such as the global resource serialization group and the console services groups, delay cleanup until the expiration of the cleanup interval. When all members finish their cleanup or when the cleanup interval elapses, MVS puts the system in a non-restartable wait state.

For example, there are two systems in a sysplex (SYS1 and SYS2). On SYS1 the VARY XCF,SYS2,OFFLINE command is issued. MVS on SYS2 notifies the group members on SYS2 that SYS2 is going to be removed from the sysplex and then waits until the members have performed cleanup, or until the cleanup interval elapses, whichever comes first, before putting the system in a wait state.

You specify the cleanup interval on the CLEANUP keyword of the COUPLE statement of the COUPLExx parmlib member or, after IPL, using the SETXCF COUPLE command.

Controlling system availability and recovery through the SFM Policy

Sysplex failure management (SFM) allows you to define a sysplex-wide policy that specifies the actions that MVS is to take when certain failures occur in the sysplex. A number of situations might occur during the operation of a sysplex when one or more systems need to be removed so that the remaining sysplex members can continue to do work. The goal of SFM is to allow these reconfiguration decisions to be made and carried out with little or no operator involvement.

Information on the SFM policy is presented in the following topics:

- [“Overview and requirements of SFM policy” on page 180](#)
- [“Planning for a status update missing condition” on page 181](#)
- [“Handling signaling connectivity failures” on page 183](#)
- [“Planning PR/SM reconfigurations” on page 186](#)
- [“Setting Up an SFM policy” on page 187.](#)

Overview and requirements of SFM policy

An SFM policy includes the following statements:

- Policy statement
- System statement(s)
- Reconfiguration statement(s)

SFM allows you to define responses for:

- Signaling connectivity failures in the sysplex
- System failures, indicated by a status update missing condition
- Reconfiguring systems in a PR/SM environment
- Signaling sympathy sickness

If the sysplex includes a coupling facility, the full range of failure management capabilities that SFM offers is available to the sysplex. For SFM to handle signaling connectivity failures without operator intervention or to isolate a failing system, a coupling facility must be configured in the sysplex.

Requirements for using the SFM policy

For a sysplex to take advantage of an SFM policy, the policy must be active on all systems in the sysplex. That is:

- All systems must be running OS/390 or MVS/ESA SP Version 5.
- An SFM policy must be started in the SFM couple data set.
- All systems must have connectivity to the SFM couple data set.

If any system loses access to the SFM couple data set, the policy becomes inactive in the sysplex. If that system regains access to the SFM couple data set, SFM automatically becomes active again in the sysplex.

Similarly, if an MVS/ESA SP Version 4 system joins a sysplex where an SFM policy is active, the policy is disabled for the entire sysplex. When that system is removed from the sysplex, SFM automatically becomes active again in the sysplex.

Relationship between SFM policy and XCF PR/SM policy

If a system with an active XCF PR/SM policy is connected to a couple data set with a started SFM policy, the XCF PR/SM policy is deactivated, regardless of whether the SFM policy is active in the sysplex.

Table 11 on page 181 summarizes what policy would be in effect for a system with an XCF PR/SM policy that is connected to the SFM couple data set, based on whether the SFM policy is started in the SFM couple data set and whether the other systems in the sysplex are connected to the SFM couple data set. Assume that all systems running OS/390 or MVS/ESA SP Version 5.

Table 11. Summary of Policies in Effect		
SFM Policy Started in Couple Data Set	All Systems Connected to Couple Data Set	Policy in Effect
No	Does not matter	XCF PR/SM Policy
Yes	No	Neither policy
Yes	Yes	SFM Policy

Planning for a status update missing condition

A status update missing condition occurs when a system does not update its status information within the time interval specified on the INTERVAL keyword in COUPLExx. SFM allows you to specify how a system is to respond to this condition. Depending on your sysplex configuration, you can specify one of the following responses on the system statement of the SFM policy: PROMPT, ISOLATETIME, RESETTIME, or DEACTTIME. IBM suggests using ISOLATETIME(0) to allow SFM to isolate and partition a failed system without operator intervention and without undue delay.

If a system loses access to the SFM couple data set, its last known action specified in the SFM policy continues to be used in response to a status update missing condition. If there is no active SFM policy in the sysplex, the default action of ISOLATETIME(0) is used in response to a status update missing condition.

Note that a system that is not updating its status information may have gone into a nonrestartable disabled wait state, or it may have failed in any of a number of other ways that do not immediately result in a disabled wait state (for example, the system may be hung, looping, or in a restartable wait state). In all cases, *regardless of the SFM policy option that you choose for handling status update missing conditions, it is highly recommended that you enable the Automatic I/O Interface Reset Facility*, which will cause an I/O interface reset to be performed if and when the system enters a nonrestartable disabled wait state. Enabling this facility will ensure that the system's I/O RESERVEs are released in a timely fashion when the system enters a nonrestartable disabled wait state. If this option is not enabled, then in some cases RESERVEs held by the unresponsive system may not be released until the system is eventually manually reset, which may result in resources being unavailable to the rest of the sysplex until this reset occurs. See the *PR/SM Planning Guide* for more information about the Automatic I/O Interface Reset Facility and how to enable it.

Prompting the operator

If PROMPT is specified, the system waits for the length of the operator notification interval (the OPNOTIFY value in COUPLExx) and then notifies the operator with message IXC402D.

As always when responding to the IXC402D prompt, it is important to take the appropriate reset action to reset the system image, and then reply to the prompt, in a timely fashion. Otherwise, resources held by the system will be unavailable to the rest of the sysplex. It is also crucial that this prompt *not* be responded to until the appropriate reset action has been taken, or data integrity problems may result. Note that the system reset action that is taken prior to responding to the prompt will cause RESERVEs held by the target system to be released.

Isolating a failing system

System isolation allows a system to be removed from the sysplex without operator intervention, while ensuring that data integrity in the sysplex is preserved. Specifically, system isolation (sometimes called "fencing") terminates all in-progress I/O activity and coupling facility accesses, and prevents any new I/O activity and coupling facility access from starting, thus ensuring that the system is unable to access and modify shared I/O resources that the rest of the sysplex is using. System isolation therefore allows the

sysplex to free up serialization resources (for example, locks and ENQs) that are held by the target system so that they may be acquired and used by the rest of the sysplex, while still preserving data integrity for all shared data.

However, note that additional steps may be required in order to ensure that any RESERVEs held by the target system are released. If the target system goes into a nonrestartable disabled wait state (either prior to, or as a result of, the system isolation action taken against it), and if the Automatic I/O Interface Reset Facility is enabled, then the interface reset that results from this will ensure that the target system's RESERVEs get released. However, if the target system does *not* go into a nonrestartable wait state, or if the Automatic I/O Interface Reset Facility is *not* enabled, the RESERVEs held by the target system may not be released. In this case, a manual reset action must be taken against the target system image in order to cause the RESERVEs to be released. *It is highly recommended that you enable the Automatic I/O Interface Reset Facility* for this reason.

System isolation requires that a coupling facility be configured in the sysplex and that the system being isolated and at least one active system have connectivity to the same coupling facility.

Also note that a system that is manually reset or re-IPLed cannot be isolated and will therefore require manual intervention to be removed from the sysplex. Therefore, to remove a system from the sysplex, it is recommended that you use the VARY XCF,sysname,OFFLINE command. If SFM is active, it will then attempt to isolate the system.

The ISOLATETIME and SSUMLIMIT SFM administrative data utility parameters indicate how long SFM will wait after detecting a status update missing condition before starting to isolate the failing system:

- If a system has not updated its status within the failure detection interval and is *not* producing XCF signaling traffic, SFM will start to isolate the failing system at the expiration of the ISOLATETIME interval. As of z/OS V1R11, ISOLATETIME(0) (isolating the failing system immediately) is the default action and interval.
- When you have specified or defaulted to SSUMLIMIT(NONE), and a system has not updated its status within the failure detection interval but continues to produce XCF signaling traffic, SFM prompts the operator to optionally force the removal of the system. The fact that XCF signalling continues indicates that the system is functional but may be experiencing a temporary condition that does not allow the system to update its status. If the operator decides that removal of the system is necessary, message IXC426D provides the prompt to isolate the system and remove it from the sysplex. In this case, the ISOLATETIME interval specified in the SFM policy is ignored.

If XCF signaling also stops, SFM will start to isolate the failing system at the expiration of the ISOLATETIME interval.

- With a value other than none specified for the SSUMLIMIT SFM administrative data utility parameter, SFM will start to isolate the system when the time specified for the SSUMLIMIT parameter has expired for a system that is in status update missing condition but still producing XCF signalling traffic.

If the system stops producing XCF signalling traffic, SFM may start to isolate the failing system before the SSUMLIMIT time expires, at the expiration of the ISOLATETIME interval.

See [“SFM parameters for the administrative data utility” on page 392](#).

If an isolation attempt is not successful (for example, if the failing system is not connected through a coupling facility to another system in the sysplex), message IXC102A prompts the operator to reset the system manually so that the removal can continue.

As always when responding to the IXC102A prompt, it is important to take the appropriate reset action to reset the system image, and then reply to the prompt, in a timely fashion. Otherwise, resources held by the system will be unavailable to the rest of the sysplex. It is also crucial that this prompt *not* be responded to until the appropriate reset action has been taken, or data integrity problems may result. Note that the system reset action that is taken prior to responding to the prompt will cause RESERVEs held by the target system to be released.

Figure 43 on page 183 shows a three-system sysplex with an active SFM policy. SYSB and SYSC are connected to a coupling facility. If either SYSB or SYSC enters a status update missing condition, the

system can be isolated by the other. However, because SYSA is not connected to the coupling facility, it cannot participate in isolation in case of failure.

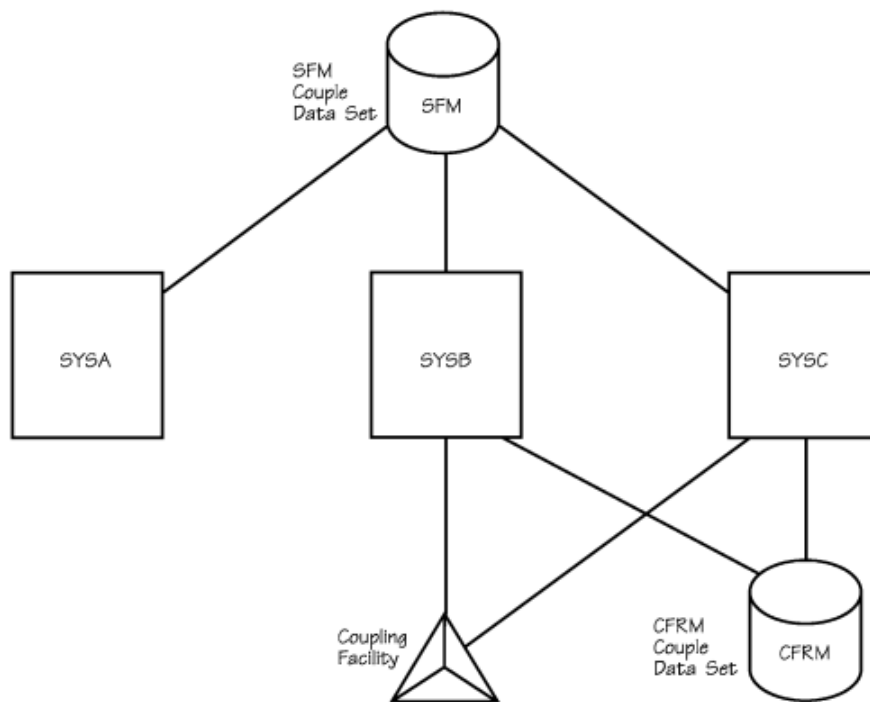


Figure 43. Three-System Sysplex with Active SFM Policy

Resetting or deactivating a failed system on PR/SM

When a system running in a PR/SM partition fails, another system in the sysplex that is running in a partition on the same processor can reset or deactivate the failed system and remove it from the sysplex without operator intervention.

The RESETTIME and DEACTTIME parameters allow you to specify how much time should elapse after the status update missing condition is detected before the action is taken.

If the reset or deactivate action against the failed system is unsuccessful, the system that initiated such action falls back to the system default of isolating the failed system immediately, as if ISOLATETIME(0) was specified.

If system isolation is possible, IBM suggests that you use the ISOLATETIME parameter, because the RESETTIME and DEACTTIME parameters forcibly terminate I/O in-progress instead of giving time to clean up. Note that the SSUMLIMIT parameter also applies to resetting or deactivating a failed system. See [“Isolating a failing system” on page 181](#).

Handling signaling connectivity failures

All systems in the sysplex must have signaling paths to and from every other system at all times. Loss of signaling connectivity between sysplex systems can result in one or more systems being removed from the sysplex so that the systems that remain in the sysplex retain full signaling connectivity to one another. SFM can eliminate operator intervention when signaling connectivity between two or more systems is lost.

The CONNFALL parameter indicates whether SFM is to handle signaling connectivity failures for the sysplex. You specify CONNFALL on the DEFINE POLICY statement of the SFM policy.

If you specify CONNFALL(YES) (the default), and there is a signaling connectivity failure, SFM automatically determines which systems to keep and which to remove from the sysplex and then attempts to implement that decision by system isolation.

In handling a system connectivity failure, SFM attempts to maximize the aggregate value of the surviving sysplex to the installation. The WEIGHT parameter of the system statement allows you to indicate the relative value of each system in the sysplex so that SFM can base its decision on installation-specified values. Ensure that you consider carefully the relative importance of each system when assigning the weight. For example, if there is a system that should never be partitioned out of the sysplex when a connectivity failure occurs, its weight should be greater than the combined weights of all other systems in the sysplex. In cases where the weights of possible surviving sysplex configurations are equal, SFM's decision on which configuration to keep is arbitrary. (If SFM is unable to determine all potential reconfigurations in a timely manner, SFM chooses the best fully-connected configuration identified to that point.)

Note that signaling connectivity failures and system failures are not necessarily detected at the same time. For some system failures, the loss of connectivity might be detected immediately but the system failure might not be detected until the failure detection interval has expired. If you have a signaling connectivity failure and an undetected system failure at the same time, SFM considers the weight of that failed system in the surviving sysplex configuration. When both a system failure and a connectivity failure are detected, the system failure is resolved first.

Figure 44 on page 184, illustrates a sysplex made up of three systems, SYSA, SYSB, SYSC. Assume that connectivity is lost between SYSA and SYSB.

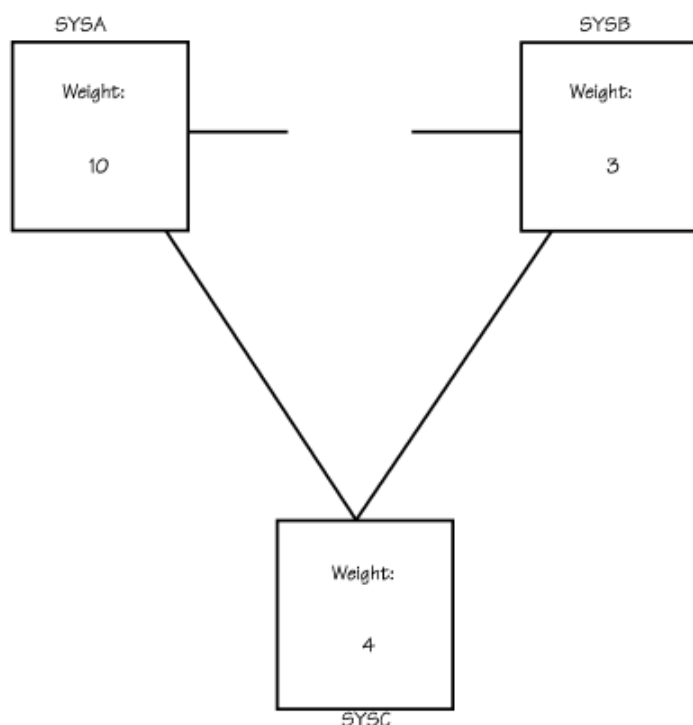


Figure 44. Signaling Connectivity Failure Between SYSA and SYSB

SFM determines that the sysplex can be reconfigured as SYSA and SYSC or as SYSB and SYSC. The new sysplex made up of SYSA and SYSC has a weight of 14, whereas the sysplex made up of SYSB and SYSC has a weight of 7. SFM chooses the sysplex with the higher weight and removes SYSB from the sysplex.

If a system that is being removed is connected to a coupling facility that is also connected to an active system, SFM can reconfigure the sysplex by isolating and removing the system without operator intervention. Without a coupling facility, a system cannot be isolated; in this case, message IXC102A prompts the operator to reset the system manually before it is removed from the sysplex.

CONNFAIL(NO) indicates that in a connectivity failure situation, MVS is to prompt the operator with message IXC409D to retry or remove the affected system(s).

Handling coupling facility connectivity failures

Loss of connectivity to a coupling facility can occur because of a failure of the coupling facility attachment or because of certain types of failures of the coupling facility itself. MVS provides the capability to initiate a rebuild of one or more structures in the coupling facility to which connectivity has been lost, using the CFRM policy and optionally, the SFM policy.

For each structure defined in the CFRM policy, you can specify a value that MVS will use to determine whether to initiate a structure rebuild when a loss of connectivity occurs. When an active SFM policy is in effect in the sysplex and a coupling facility loss of connectivity occurs, MVS uses the system weights defined in the SFM policy in conjunction with the REBUILDPERCENT value defined in the CFRM policy to determine whether to initiate a structure rebuild. Using the SFM policy system weights, MVS calculates the weighted percentage of lost connectivity and compares the result with the user-specified REBUILDPERCENT value. If the calculated percentage value is greater than or equal to that specified by REBUILDPERCENT, MVS will initiate the structure rebuild.

With OS/390 R3 and higher, with OW30814 installed, MVS will initiate a rebuild when loss of connectivity to a structure occurs and there is no active SFM policy in the sysplex.

See [“Specifying a rebuild threshold” on page 59](#) for information about MVS-initiated structure rebuild.

Handling signaling sympathy sickness

Signaling sympathy sickness can occur when a stalled member on the target system consumes too many I/O buffers and causes the sending system to run out of outbound I/O buffers. The MEMSTALLTIME parameter in the SFM policy allows SFM to take automatic action to alleviate signaling sympathy sickness caused by a stalled XCF group member. This is especially useful in cases where the stall condition is so severe that both manual intervention and system automation are impeded. If you specify MEMSTALLTIME(seconds), XCF will terminate the stalled member that is consuming the greatest number of applicable I/O buffers after sympathy sickness is declared and persists for MEMSTALLTIME seconds. If the sympathy sickness condition continues to persist, XCF will continue to terminate stalled members in this fashion until the problem is resolved.

If a stalled member is to be terminated to resolve a signaling sympathy sickness problem, XCF terminates the associated task or space that the member identified at join time. Such termination of the task or space has the same effect on the member as would the demise of that task or space through any other failure, and therefore suitable cleanup and/or recovery should be expected. There is some risk that the exploiter might not deal well with such termination. However, failure to resolve the signaling sympathy sickness problem in a timely manner may well lead to the demise of one or more systems.

XCF messages

The following messages can help you understand how the sysplex can handle sympathy sickness that is caused by one or more stalled or impaired XCF members.

IXC430I, IXC431I, and IXC432I: The system issues the following messages to provide information about the current state of the stall for a particular group member on that system:

- **IXC430I.** The system issues this message to the operator console to indicate that ts one or more XCF group members that are stalled on the system.
- **IXC431I and IXC432I.** The system might issue these messages periodically to the log. IXC431I indicates which XCF group members are stalled on the system, and IXC432I indicates when the stalled condition is alleviated. XCF might issue abend x00C reason x020F0006 to initiate internal XCF self verification and other actions to address the stall. The abend does not directly impact the stalled application. If an internal XCF problem is discovered, a dump is taken. The system creates an entry in logrec to document the situation even if a dump is not taken.

IXC440E, IXC631I, and IXC632I: The system issues the following messages when another system has one or more members of an XCF group that is not processing their XCF work in a timely manner and the stall is having an impact on the local system:

- IXC440E identifies the system that has one or more members of an XCF group that is not effectively processing their XCF work and the system in the sysplex that any of those members are affecting. Action must be taken on the system with the stalled or impaired XCF members to resolve the problem and avoid further sympathy sickness.
- For each stalled XCF member that IXC440E identifies, the system issues IXC631I to provide further information about the critical situation on the affected system; for example, the system might be in the process of being removed from the sysplex.
- IXC632I indicates when the stalled condition is alleviated.

IXC640E: The system might be able to resolve the problem automatically if the SFM policy permits XCF to take action. Look for message IXC640E that identifies the affected system and how XCF is permitted to act. On many occasions the system successfully resolves the situation during the course of normal processing.

IXC640E indicates either that the system has at least one XCF group member that appears to be stalled and is not processing its XCF work in a timely manner or that the system has at least one critical XCF group member that appears to be impaired. If you have MEMSTALLTIME enabled for the SFM policy, the policy can specify that action is to be taken for the system. If you do not have an SFM policy or if MEMSTALLTIME is not enabled, the operator must take action to correct the situation.

If the impact persists and the active SFM policy MEMSTALLTIME specification for the local system allows XCF to take action to address the problem, and the impacted system is not being removed from the sysplex, XCF terminates the indicated member or removes its system from the sysplex. If the impacted system is being removed from the sysplex when XCF determines whether it should take action against the member, XCF does not consider this a sympathy sickness impact. In particular, if the indicated system is the only one that is impacted, no action is taken against the member because the sympathy sickness disappears after the system is removed from the sysplex. If XCF terminates the member, it requests a dump and issues message IXC615I.

Note that members of the SYSGRS, SYSMCS, and SYSXCF groups are not directly terminated. Instead, the system on which the members reside is removed from the sysplex with wait-state 0A2 reason code 160. See [“Signaling sympathy sickness” on page 130](#) for more information about signaling sympathy sickness.

Planning PR/SM reconfigurations

After a system running in a PR/SM partition is removed from the sysplex, SFM allows a remaining system in the sysplex to reconfigure processor storage for use by the remaining systems.

You specify PR/SM reconfiguration statements on the RECONFIG statement of the SFM policy. PR/SM reconfiguration definitions include the name of the failing system (FAILSYS), the name of the system that is to perform the reconfiguration actions (ACTSYS), and whether PR/SM is to deactivate a specific system or all logical partitions in the addressing range of the acting system (TARGETSYS). You can also specify whether the system performing the reconfiguration is to take over the target system's storage and/or expanded storage. To carry out a PR/SM reconfiguration, ACTSYS and TARGETSYS must be on the same processor. FAILSYS can be the same system as TARGETSYS, or it can be another system, on the same processor or on another processor.

When the TARGETSYS is deactivated, its CPU resources are added to the pool of shared CPU resources, thereby increasing the CPU cycles available to all shared logical partitions. The distribution of the additional CPU cycles is proportional to the weights of the active shared logical partitions. When the TARGETSYS is reactivated, its CPU resources are removed from the shared CPU resource pool.

Figure 45 on page 187 shows a three-system sysplex, including two MVS images (SYSA and SYSB) running in logical partitions on a single processor and one MVS (SYSC) on a single processor. Because an SFM policy is active, XCFPOLxx specifications are not active. SYSB can be configured to take over the resources associated with the logical partition of SYSA if either SYSA or SYSC is removed from the sysplex. This would give SYSB the extra resources needed to handle the work of either SYSA or SYSC.

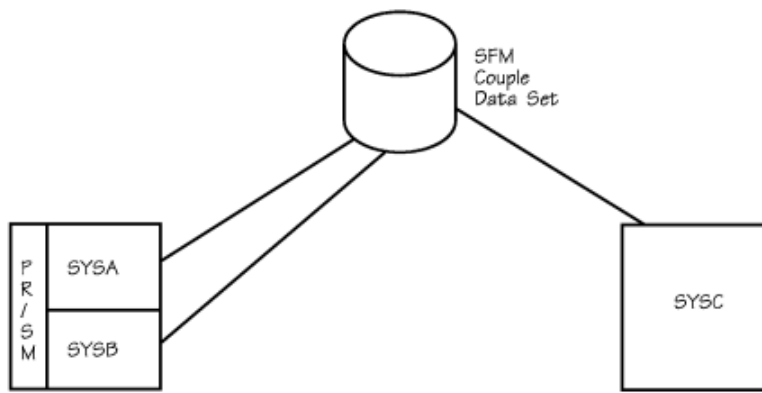


Figure 45. Three-System Sysplex on Two Processors with Active SFM Policy

Note: PR/SM reconfiguration actions can take place only if XCF can do a system reset of the MVS system and its LPAR and deactivate the LPAR.

For information about the hardware PR/SM feature and the assignment of LPARs, central processors (CPs), and storage elements, see *PR/SM Planning Guide*.

Setting Up an SFM policy

To implement an SFM policy, you need to:

- Format an SFM couple data set and ensure that it is available to all systems in the sysplex
- Define the SFM policy
- Start the SFM policy in the SFM couple data set.

Information on formatting and defining a couple data set is included in [“Considerations for function couple data sets”](#) on page 37.

Defining SFM policies

The administrative data utility, IXCMIAPU, described in [Chapter 12, “Administrative data utility,”](#) on page 335 allows you to associate the definitions with a policy name and to place the policy in a pre-formatted SFM couple data set.

Starting an SFM policy

To start an SFM policy (POLICY1, for example) that is defined in the SFM couple data set, issue:

```
SETXCF START,POLICY,POLNAME=POLICY1,TYPE=SFM
```

Updating an SFM policy

For information on updating the SFM policy dynamically, see [“Updating policies in a sysplex”](#) on page 38.

Controlling availability and recovery through XCFPOLxx (PR/SM only)

The SFM policy includes all the function available through XCFPOLxx. If a system is connected to a couple data set with a started SFM policy, all XCFPOLxx specifications on that system are deactivated, regardless of whether the SFM policy is active in the sysplex.

Because the SFM policy provides function beyond that provided by the XCF PR/SM policy, it is generally recommended that you use the SFM policy to manage failures in your sysplex. However, in cases where

you cannot activate an SFM policy (for example, if the sysplex includes an SP Version 4 system), activating an XCF PR/SM policy can be useful.

For a description of the functions provided through XCFPOLxx, see [“Resetting or deactivating a failed system on PR/SM” on page 183](#) and [“Planning PR/SM reconfigurations” on page 186](#).

Implementing XCFPOLxx

You specify XCFPOLxx values in parmlib, activate (or deactivate) the policy with the SETXCF PRSMPOLICY command, and display the name of the active policy with the DISPLAY XCF,PRSMPOLICY command.

To change an existing XCF PR/SM policy or activate a new policy, issue:

```
SETXCF PRSMPOLICY,ACTIVATE=memname
```

where memname is the parmlib member that contains the new XCF PR/SM policy.

To deactivate the XCF PR/SM policy, issue:

```
SETXCF PRSMPOLICY,DEACTIVATE
```

When command processing completes, message IXC322I or IXC321I is issued.

Table 12 on page 188 shows the corresponding statements and keywords, in the SFM policy and the XCF PR/SM policy, for initiating PR/SM reconfigurations. For example, the FAILSYS keyword in the SFM policy corresponds to SYSGONE in XCFPOLxx.

Table 12. Statements and Keywords for Initiating PR/SM Reconfigurations	
SFM Policy	XCFPOLxx
RECONFIG	No statement
FAILSYS	SYSGONE
ACTSYS	SYSTEM
TARGETSYS	DEACTIVATE

Examples

These examples illustrate the use of the SFM policy to handle failure conditions in a sysplex. Where possible, corresponding XCFPOLxx specifications are also provided.

Example 1: Failure management in a PR/SM environment — one processor

Figure 46 on page 188 shows two MVS systems, PRIMARY and BACKUP1. The two systems are the only systems in LPARs on the same PR/SM processor and in the same sysplex. In the example, the group members of the multisystem applications on PRIMARY are using more processor storage resources than their group members on BACKUP1.

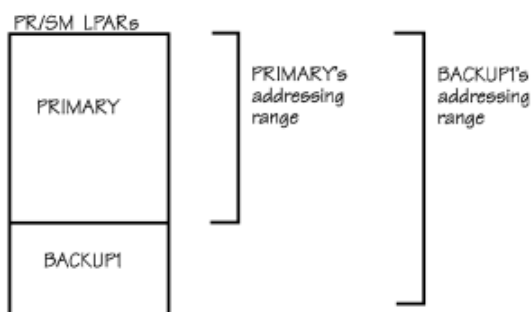


Figure 46. Failure Management in a PR/SM Environment — One Processor

Objective

If PRIMARY fails, enable an application's group members on BACKUP1 to take over the workload of their group members on PRIMARY without operator intervention. This requires more processor storage resources than is available to BACKUP1, so MVS must obtain more storage.

Note that you must define the processor storage to the LPARs in such a way that the BACKUP1 LPAR can obtain the storage freed from PRIMARY. For information on how to do this, see *PR/SM Planning Guide*.

SFM policy specifications

```
DEFINE POLICY NAME(POL1) CONNFALL(NO)

  SYSTEM NAME(PRIMARY)
    RESETTIME(5)

  RECONFIG
    FAILSYS(PRIMARY) ACTSYS(BACKUP1) TARGETSYS(ALL)
    STORE(YES) ESTORE(YES)
```

Scenario

If PRIMARY stops updating its status, the following events occur, based on the policy.

- When the failure detection interval specified on the INTERVAL keyword in PRIMARY's COUPLExx parmlib member elapses, MVS on BACKUP1 detects a status update missing condition for PRIMARY.
- BACKUP1 waits 5 seconds. If the status update missing condition on PRIMARY is not resolved, BACKUP1 performs a system reset of PRIMARY.
- BACKUP1 notifies group members on remaining systems in the sysplex (only BACKUP1 in the example) that PRIMARY has failed and no longer has access to shared resources.
- Group members of the multisystem applications on BACKUP1 can now take over the workload of their group members on PRIMARY.
- BACKUP1 deactivates all other LPARs in its addressing range (only PRIMARY in this example) and then acquires both the central and expanded storage assigned to those LPARs (only PRIMARY in this example).
- BACKUP1 brings all processor storage in its addressing range online. The storage previously used by PRIMARY is now available to BACKUP1. Also, the shared central processors used by PRIMARY are available to BACKUP1.
- As a result of the SFM policy, there is enough processor storage available on BACKUP1 for group members of the multisystem applications on BACKUP1 to take over the workload of their group members on PRIMARY.

Note that CONNFALL(NO) is specified on the policy statement because a coupling facility is not available.

Corresponding XCFPOLxx specifications

You can use the following XCFPOLxx specifications to achieve the same function.

```
NOSTATUS(PRIMARY) RESETTIME(5)

SYSGONE(PRIMARY) SYSTEM(BACKUP1) DEACTIVATE(ALL)
STORE(YES) ESTORE(YES)
```

Example 2: Failure management in a PR/SM environment — two processors and a coupling facility

Figure 47 on page 190 shows three MVS systems called PRIMARY, TEST, and BACKUP2. PRIMARY is on one processor and TEST and BACKUP2 are in LPARs on a PR/SM processor. PRIMARY and BACKUP2 are in the same sysplex and are connected through the same coupling facility. TEST is not in the sysplex. In the

example, the group members of the multisystem applications on PRIMARY need more processor storage resources than their group members on BACKUP2.

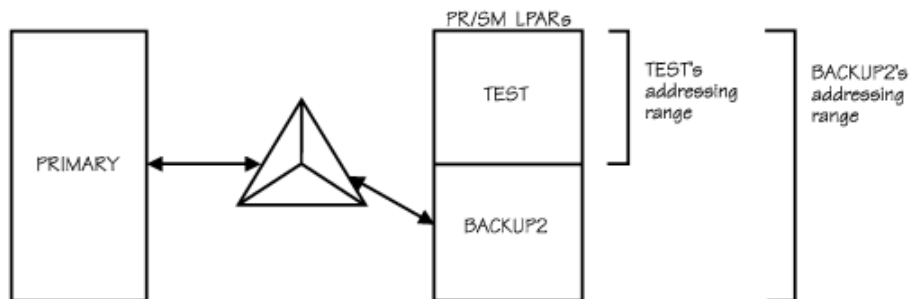


Figure 47. Failure Management in a PR/SM Environment — Two Processors and a Coupling Facility

Objective

If PRIMARY fails, enable an application's group members on BACKUP2 to take over the workload of their group members on PRIMARY. This requires more processor storage than is available to BACKUP2, so MVS obtains more storage by taking TEST down and configuring its storage online to BACKUP2.

Note that you must define the processor storage to the LPARs in such a way that the BACKUP2 LPAR can obtain the storage freed from TEST. For information on how to do this, see *PR/SM Planning Guide*.

SFM specifications

```
DEFINE POLICY NAME(POL2)
  SYSTEM NAME(PRIMARY) ISOLATETIME(5)
  RECONFIG
    FAILSYS(PRIMARY) ACTSYS(BACKUP2) TARGETSYS(ALL)
    STORE(YES) ESTORE(YES)
```

Scenario

If PRIMARY stops updating its status, the following events occur, based on the SFM policy.

- BACKUP2 isolates PRIMARY and removes it from the sysplex. Because PRIMARY and BACKUP2 are connected through the same coupling facility, this can be done without operator intervention.
- BACKUP2 then deactivates all other LPARs in its addressing range, which includes TEST.
- BACKUP2 brings online all processor storage in its addressing range. The storage previously used by TEST is now available to BACKUP2. Also, the shared central processors used by TEST are available to BACKUP2.
- Because BACKUP2 has sufficient processor storage, group members of the multisystem applications on BACKUP2 can take over the workload of their group members on PRIMARY.

Corresponding XCFPOLxx specifications

You can use the following XCFPOLxx specifications to achieve the same function:

```
SYSGONE(PRIMARY) SYSTEM(BACKUP2) DEACTIVATE(ALL)
  STORE(YES) ESTORE(YES)
```

Using the XCF PR/SM policy, the operator must respond to a prompt to remove PRIMARY before reconfiguration can begin.

Example 3: Failure management in a PR/SM environment — one processor

Figure 48 on page 191 shows two MVS systems, SYSA and SYSB. Both systems are in LPARs on the same PR/SM processor and in the same sysplex. The group members of the multisystem applications on SYSA share resources with their group members on SYSB. Group members on either system can take over the workload of the other system without additional processor storage



Figure 48. Failure Management in a PR/SM Environment — One Processor

Objective

If either system fails, without operator intervention, system reset the failing system and notify the application's group members on the other system of the failure.

SFM specifications

```
SYSTEM NAME(SYSA)
      RESETTIME(20)
SYSTEM NAME(SYSB)
      RESETTIME(10)
```

Scenario

If SYSA stops updating its status, the following events occur, based on the policy.

- When the failure detection interval specified on the INTERVAL keyword in SYSA's COUPLExx parmlib member elapses, SYSB detects a status update missing condition for SYSA.
- SYSB waits 20 seconds. If the status update missing condition on SYSA is not resolved, SYSB performs a system reset of SYSA.
- Then, SYSB notifies the group members of the multisystem applications on SYSB that SYSA has failed. Thus, group members on SYSB know that their group members on SYSA are no longer using shared resources and that they must take over their workload.

Corresponding XCFPOLxx specifications

You can use the following XCFPOLxx specifications to achieve the same function:

```
NOSTATUS(SYSA) RESETTIME(20)
NOSTATUS(SYSB) RESETTIME(10)
```

Example 4: Failure management with SFM policy and a coupling facility

Figure 49 on page 192 shows five MVS systems, SYSA, SYSB, SYSC, SYSD, and SYSE in the same sysplex. All systems have connectivity to the same coupling facility. SYSD and SYSE are running in logical partitions on the same processor.

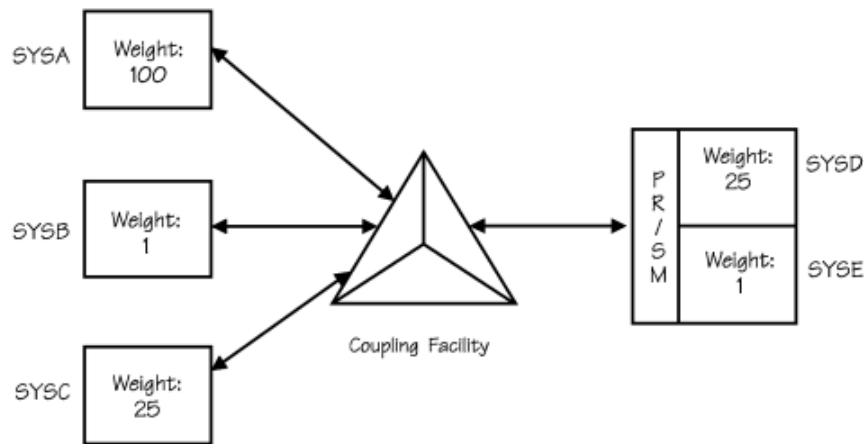


Figure 49. Failure Management with SFM Policy and a Coupling Facility

Objective

This policy defines actions to be taken for the following conditions:

- System failures
- Signaling connectivity failures
- PR/SM reconfiguration actions when a system is removed from the sysplex.

SFM Policy specifications

```
DEFINE POLICY NAME(POL4) CONNFAIL(YES)

  SYSTEM NAME(*) /* Policy default statement */
    ISOLATETIME(0)

  SYSTEM NAME(SYSA)
    PROMPT
    WEIGHT(100)

  SYSTEM NAME(SYSC)
    WEIGHT(25)

  SYSTEM NAME(SYSD)
    WEIGHT(25)

  RECONFIG
    FAILSYS(SYSE)
    ACTSYS(SYSD)
    TARGETSYS(SYSE)
    STORE(YES) ESTORE(YES)
```

Scenario

- Connectivity failure

The policy specifies one heavily weighted system, SYSA. The policy also specifies two lesser weighted systems, SYSC and SYSD. Two systems, SYSB and SYSE, will use the system default weight of 1, since they do not have a system statement and weight is not specified in the policy default statement. (For a description of how SFM uses weights to make isolation decisions, see [“Handling signaling connectivity failures”](#) on page 183.)

- System failure

If SYSA stops updating its status, the operator will be prompted, and no automated action will take place. Because SYSB and SYSE do not have a system statement and the system statements for SYSC

and SYSD do not specify an action for a status update missing condition, if SYSB, SYSC, SYSD, or SYSE enters a status update missing condition, the policy default action will be used, and a system isolation will be attempted with no delay.

- Reconfiguration

If SYSE is removed from the sysplex for any reason, and SYSD is active, then SYSD will acquire both the central and expanded storage assigned to SYSE.

Controlling job availability and recovery through automatic restart management

Automatic restart management is an MVS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which it is running fails, automatic restart management can restart the job or task without operator intervention.

The goals of SFM and automatic restart management are complementary. While SFM keeps the sysplex running, automatic restart management keeps specific work in the sysplex running. If a job or task fails, automatic restart management restarts it on the same system it was running on at the time of the failure. If a system fails, automatic restart management restarts the work on other systems in the sysplex; this is called a **cross-system restart**.

A program cannot use both automatic restart management and checkpoint/restart. If a program using checkpoint/restart tries to register with automatic restart management, the request is rejected.

Requirements for participating in automatic restart management

To participate in automatic restart management,

- Your system must be running supported versions of z/OS and JES2 or JES3.
- A system must be connected to an ARM couple data set with an active automatic restart management policy.
- A batch job or started task must register with automatic restart management. (That is, it must issue the IXCARM macro. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on the IXCARM macro.)

Note: A batch job or started task registered with automatic restart management can only be restarted within the same JES XCF group. (That is, JES2 MAS or JES3 complex.)

- The XCF address space must have adequate authorization through the z/OS Security Server (RACF) or an equivalent security product.

In order to perform restarts, ARM must issue operator commands from the XCF address space (XCFAS). You must be certain that the XCFAS has enough authority to issue any commands required to restart a failed element. If you are using ARM, IBM recommends that your RACF support personnel define the XCFAS in the RACF STARTED class or the started procedures table with the trusted attribute. See [z/OS Security Server RACF Security Administrator's Guide](#) for information about using the STARTED class. See [z/OS Security Server RACF System Programmer's Guide](#) for information about using the started procedures table.

Using automatic restart management

IBM provides policy defaults for automatic restart management, as shown in [Table 13 on page 194](#). You can use these defaults or override them for your installation.

Automatic restarts do not need to be enabled at all times. For example, you might not want automatic restart management enabled for restarts unless certain jobs are running, or during off-shift hours. However, even while automatic restarts are disabled, elements can register with automatic restart management as long as the system is connected to an ARM couple data set.

To implement the policy, you need to:

- Format an ARM couple data set and ensure that it is available to all systems in the sysplex that are to participate in automatic restart management. (That is, all systems running batch jobs or started tasks that might be restarted by automatic restart management and all systems on which batch jobs or started tasks might be restarted if the system on which they are running fails.)

Information on formatting and defining a couple data set is included in [“Considerations for function couple data sets”](#) on page 37.

- Start the automatic restart management policy in the ARM couple data set. Once the policy is started, all systems connected to the ARM couple data set will use the same active policy.

To start automatic restart management with the policy defaults, issue:

```
SETXCF START,POLICY,TYPE=ARM
```

To start your own automatic restart management policy (mypol, for example), issue:

```
SETXCF START,POLICY,TYPE=ARM,POLNAME=mypol
```

Use the SETXCF STOP command to disable automatic restarts.

- For information on updating the automatic restart management policy dynamically, see [“Updating policies in a sysplex”](#) on page 38.

Customizing the automatic restart management policy

You can override one or more of the policy defaults through the administrative data utility (IXCMIAPU), described in [Chapter 12, “Administrative data utility,”](#) on page 335, through the IXCARM macro, or through the exits.

For example, your installation might want to define a policy to group together elements (such as CICS and DB2) that are related or perform interdependent processing and, therefore, must run on the same system. These groups are referred to as **restart groups**.

To determine whether you need to override any policy defaults, you must decide how you want to manage automatic restarts and whether the defaults satisfy those goals. Use the following table and the considerations that follow. The table provides a list of the automatic restart management parameters of IXCMIAPU, their default values (which you can override in the policy), and the IXCARM macro parameters that can override some of the defaults.

Table 13. Automatic Restart Management Parameters for IXCMIAPU and IXCARM Parameter Overrides			
IXCMIAPU Parameter	Optional or Required	Default	IXCARM Parameter that Overrides Default
RESTART_ORDER	Optional	LEVEL(2),SYSLVL2	None
LEVEL	Optional	(2)	None
ELEMENT_NAME	Optional	None	None
ELEMENT_TYPE	Optional	(SYSLVL2)	ELEMTYPE
RESTART_GROUP	Required	DEFAULT	RESTART_GROUP
TARGET_SYSTEM	Optional	(*)	TARGET_SYSTEM
FREE_CSA	Optional	(0,0)	FREE_CSA
RESTART_PACING	Optional	(0)	RESTART_PACING
ELEMENT	Required	None	None
RESTART_ATTEMPTS	Optional	(3,300)	None
RESTART_TIMEOUT	Optional	(300)	None
READY_TIMEOUT	Optional	(300)	None

Table 13. Automatic Restart Management Parameters for IXCMIAPU and IXCARM Parameter Overrides (continued)

IXCMIAPU Parameter	Optional or Required	Default	IXCARM Parameter that Overrides Default
TERMTYPE	Optional	(ALLTERM)	TERMTYPE
RESTART_METHOD	Optional	(BOTH,PERSIST)	STARTTXT

Considerations

The following considerations will help you determine the policy information you need for each batch job or started task that will register as an element of automatic restart management. The policy parameters you would use to override the defaults are shown in parentheses. For more information on these parameters, see [“Automatic restart management parameters for administrative data utility” on page 338](#).

1. Determine which batch jobs and started tasks will be using automatic restart management for recovery purposes. For the IBM products that use automatic restart management, read their documentation for any policy considerations.
2. Determine if any of these elements is interdependent — that is, need to run on the same system. (RESTART_GROUP).

Note that any elements that are not explicitly assigned to a restart group become part of the restart group named DEFAULT. Thus, if these elements are restarted, they are restarted on the same system.
3. Determine if there is an order in which MVS should restart these elements — if any elements in the restart group are dependent upon other elements being restarted and ready first (RESTART_ORDER).
4. Determine if the elements in a restart group need to be restarted at specific intervals (RESTART_PACING).
5. Determine if the element should be restarted when only the element fails, or when either the element or the system fails (TERMTYPE).
6. Determine whether specific JCL or command text is required to restart an element (RESTART_METHOD).
7. Determine if a minimum amount of CSA/ECSA is needed on the system where the elements in a restart group are to be restarted (FREE_CSA).
8. Determine if you want the elements in the restart group to be restarted on a specific system (TARGET_SYSTEM).

Consider requirements like:

- Which systems have access to the ARM couple data sets.
 - Which systems run within the same JES XCF group.
 - The workloads of the systems in the sysplex, and how they might be affected if these jobs were restarted.
 - CPU capacity needed.
 - DASD requirements.
 - Which systems have the class of initiator required by batch jobs that might need to be restarted on another system.
9. Determine if an element should be restarted and, if so, how many times it should be restarted within a given interval (RESTART_ATTEMPTS). If an element should not be restarted, set RESTART_ATTEMPTS to 0.
 10. Determine how long MVS should wait for an element to re-register once it has been restarted (RESTART_TIMEOUT).
 11. Determine how long MVS should wait for an element to indicate it is ready to work after it has been restarted (READY_TIMEOUT).

Using exits to customize automatic restart management processing

Several exits can influence how an element will be restarted:

- To coordinate restarts with other automation packages, use the element restart exit (IXC_ELEM_RESTART) described in *z/OS MVS Installation Exits*.
- To prepare a system that will be receiving additional workload from a failing system, use the workload restart exit (IXC_WORK_RESTART), described in *z/OS MVS Installation Exits*.
- To prepare a system for the restart of an element or to prevent the restart of an element, use the event exit, described in *z/OS MVS Programming: Sysplex Services Guide*.

Using the system status detection partitioning protocol and BCPii for availability and recovery

The System Status Detection (SSD) partitioning protocol exploits BCPii interfaces to use z/Series hardware services to discover the status of failed systems in a sysplex during sysplex recovery processing. Using BCPii, XCF can bypass specified intervals such as the failure detection interval (FDI), the SSUM ACTION interval, and the cleanup interval; and it can avoid fencing processing and manual intervention requirements to shorten sysplex recovery processing time. To enable XCF to use the SSD partitioning protocol and BCPii in the sysplex, meet the following requirements:

- The XCF address space must have adequate authorization through the z/OS Security Server (RACF) or an equivalent security product to access BCPii defined FACILITY class resources. See topic "Assigning the RACF TRUSTED attribute" in *z/OS MVS Initialization and Tuning Reference* for information about using RACF to assign the TRUSTED attribute to the XCF address space.
- The BCPii interfaces to invoke zSeries Hardware APIs must be enabled. See the "BCPii Setup and Installation" topic in *z/OS MVS Programming: Callable Services for High-Level Languages* for information about installation and configuration steps and SAF authorization requirements to enable BCPii to invoke zSeries Hardware APIs.
- All systems in the sysplex must operate on z10 EC GA2, z10 BC GA1, or newer hardware and be at z/OS V1R11 or higher to take full advantage of the functionality associated with the system status detection partitioning protocol. At a minimum, all systems in the sysplex must be at z/OS V1R11 or have installed toleration PTFs for OA26037 on V1R10 and V1R9 systems in the sysplex to use a sysplex couple data set formatted using the ITEM NAME(SSATDET) NUMBER(1) control statement, which is required by the protocol.

A system running on z/OS V1R11 (or higher) and on the requisite level of hardware and enabled to use the SSD partitioning protocol can exploit the full functionality of the SSD partitioning protocol. Full functional exploitation means that a system can be targeted by other systems in the sysplex enabled to use the SSD partitioning protocol for sysplex recovery, and that the system can target other systems running on z/OS V1R11 (or higher) and the requisite level of hardware in the sysplex to expedite sysplex recovery when the targeted system of recovery processing becomes demised.

A system running on z/OS V1R11 (or higher) and down-level hardware is only eligible to target other systems that are enabled to exploit the full functionality of the SSD partitioning protocol. A system not running on the requisite hardware can not be the target of SSD partitioning protocol functions.

A system running on down-level hardware, a down-level version of z/OS, or running as a VM Guest cannot be enabled to use the SSD partitioning protocol.

- The sysplex couple data set must be formatted using the ITEM NAME(SSATDET) NUMBER(1) control statement. See "Coding the couple data set format utility" on page 314 for information about specifying the ITEM NAME(SSATDET) NUMBER(1) control statement and Chapter 3, "Planning the couple data sets," on page 27 for information about sysplex couple data set planning.
- The SYSSTATDETECT function must be enabled. By default, the SYSSTATDETECT function is enabled. The current setting of the SYSSTATDETECT function can be determined by issuing a DISPLAY XCF,COUPLE command.

See *z/OS MVS Initialization and Tuning Reference* for information about specifying SYSSTATDETECT in the COUPLExx parmlib member, and *z/OS MVS System Commands* for information about using the SETXCF FUNCTIONS command to enable and disable the SYSSTATDETECT function.

With a PTF for APAR OA54778, the SSD partitioning protocol will take immediate sysplex system recovery action against a system when the CPC image that a system is running on is found to be in the "not operating" state. A CPC image is in the "not operating" state when all logical CP's assigned to an LPAR are in a stopped state. The "not operating" state may be due to a permanent reason such as a PU check stop (Logical Partition System Check Stop) or a transient condition such as when the STOP task is selected at the HMC or a restartable wait state.

For scenarios where operational actions are taken to put an LPAR into the "not operating" state in a transient way and there is intent to restart the image (such as for a TEST system) without triggering unwanted SFM/SSD/sysplex partitioning actions, these steps are recommended:

- Take action to temporarily disable the XCF/SSD partitioning protocol for the image in question. Issue the **SETXCF FUNCTIONS, DISABLE=SYSSTATDETECT** operator command.
- Based on the time the processors will be STOPPED and the LPAR will be in the "not operating" state, take action necessary to disable SFM actions and/or fencing actions against the target image once the FDI expires. These actions might include changing the FDI for the image, or changing the SFM action from **ISOLATETIME** to **PROMPT**.
- Stop the processors as needed.
- Restart the processors as needed.
- Restore the SFM actions.
- Re-enable the XCF/SSD partitioning protocol for the image by issuing the **SETXCF FUNCTIONS, ENABLE=SYSSTATDETECT** operator command.

Handling concurrent system and couple data set failures

It is possible to have concurrent loss of couple data sets, systems, and other sysplex resources if a disaster, such as a power failure, occurs. Concurrent failure of one or more couple data sets and one or more systems requires a special recovery procedure. In the event that a concurrent failure occurs, XCF issues message IXC256A. If the systems identified in message IXC256A are in an unrecoverable state (such as a disabled wait state or a loss of power), those systems must be removed from the sysplex to allow couple data set recovery to proceed.

Remove each of the systems identified in message IXC256A as follows:

1. Perform a system reset on each system identified in IXC256A.
2. From an active system in the sysplex, issue the following command, where sysname is the name of the system identified in IXC256A.

```
VARY XCF,sysname,OFFLINE
```

3. From the same active system used in the preceding step, issue the following command, where sysname is the name of the sysname used in the preceding step.

```
VARY XCF,sysname,OFFLINE,FORCE
```

Planning disaster recovery actions

A common disaster recovery solution is to use Remote Copy technologies to replicate data on DASD volumes, where all the critical volumes from the primary site are mirrored to volumes at the disaster site. Among the critical volumes which can be mirrored in this way are those containing the Sysplex and CFRM couple data sets. Some special considerations apply to the volumes containing these couple data sets.

IBM strongly recommends not to use synchronous remote copy technologies (such as peer-to-peer remote copy, or PPRC) for volumes that contain the Sysplex, CFRM couple data sets or both the Sysplex and CFRM couple data sets. Specifically, a DASD "freeze" against volumes containing these couple data

sets could fatally interfere with the ability of the sysplex to recover and reconfigure itself during a site failure, connectivity failure, or whatever problem it was that triggered the DASD "freeze" to occur in the first place. The sysplex must be able to reconfigure itself when necessary, and during synchronous remote copy failover processing, this is not possible.

IBM also recommends against the use of asynchronous remote copying technologies such as Global Mirror for managing couple data sets. Asynchronous copy transfers storage unit track images that do not necessarily correspond to the logical records defined within the couple data set. After failover to the remote site with which the copy was established, it may be impossible to IPL the sysplex using the copied couple data sets.

Instead, IBM recommends that you predefine and initialize new instances of couple data sets to be used for sysplex initialization at the disaster recovery site. In this case, a different COUPLExx parmlib member should also be defined for use at the disaster recovery site, which contains the couple data set names to be used at the disaster recovery site, and the name of the CFRM policy that is to be activated for use at the disaster recovery site.

Note:

1. When initializing the sysplex with freshly-formatted couple data sets, the CFRMPOL keyword in the COUPLExx parmlib member can be used to identify the desired CFRM policy that is to be activated automatically when the sysplex is initialized at the disaster recovery site.
2. In general, the CFRM policy that is used at the disaster recovery site is not the same policy that is normally used for production. For example, a different set of coupling facility images, with unique physical identification information, exist for use at the disaster recovery site rather than exist in the production site; the CFRM policy used at the disaster recovery site needs to define those coupling facility images for use, not the production CFs. The set of required structure definitions might also be different when running at the disaster recovery site. You can either define two totally separate CFRM policy definitions for this (one for use at the production site, and one for use at the disaster recovery site), or define one "overdefined" CFRM policy for use at both sites, which encompasses all the CF definitions and CF structure definitions that are applicable to both sites.
3. When IPLing with freshly-formatted sysplex couple data sets, operator intervention may be necessary at several points during initialization of the first system. Some applications retain configuration information in the user data portion of the records associated with their groups in the sysplex couple data set. This information will not be available when IPLing with new couple data sets, and may have to be reestablished by interaction with the operator.
4. GDPS customers should refer to GDPS procedures for couple data set management, particularly for the Logger couple data set.

Important!

Great care should be taken when replicated Sysplex or CFRM couple data sets are used to initialize a sysplex for disaster recovery testing, whenever those couple data sets are replicated by any of the following means:

- Flash copy
- Synchronous remote copy
- Asynchronous remote copy
- Having the primary couple dataset located in one site and the secondary couple data set in the other site
- Using an "old" couple dataset that was formerly in use in the production sysplex

The replicated couple data sets will contain all of the internal sysplex information from the production sysplex, which includes the following:

- The identical sysplex name
- Active system names
- Coupling facility names
- Resource ownership information

When a sysplex is initialized at the disaster recovery site using these replicated couple data sets, that sysplex will work as if it is the production sysplex. If there is physical connectivity between the disaster recovery site and the production site that allows this new instance of the sysplex to access the physical resources of the production sysplex which are still being used by the production sysplex, serious problems can result as both sysplexes erroneously attempt to share and manage the same physical resources, such as coupling facilities and CF structures.

Such problems cannot occur when the sysplex that is initialized at the disaster recovery site uses freshly-formatted Sysplex and CFRM couple data sets, and uses the CFRMPOL keyword in the COUPLExx parmlib member. The CFRMPOL keyword in the COUPLExx parmlib member can automatically initialize a disaster-recovery-only version of the CFRM policy that only defines and references resources that are valid in the disaster recovery site instead of the production site.

Figure 50 on page 199 shows an example of using the CFRMPOL keyword in the COUPLExx parmlib member.

```
COUPLE      SYSPLX(&SYSPLX.)
            CFRMPOL(CFPOLD)
            PCOUPLE(SYS1.XCFCDS.PRI)
            ACOUPLE(SYS1.XCFCDS.ALT)
            INTERVAL(25)
            CLEANUP(10)
            MAXMSG(2500)

CLASSDEF    CLASS(SMALL) CLASSLEN(956) GROUP(UNDESIG)
CLASSDEF    CLASS(DEFAULT) CLASSLEN(4028) GROUP(UNDESIG)
CLASSDEF    CLASS(LARGE) CLASSLEN(16316) GROUP(UNDESIG)

PATHOUT     STRNAME(IXCSTR1) CLASS(DEFAULT)
PATHOUT     STRNAME(IXCSTR2) CLASS(SMALL)
PATHOUT     STRNAME(IXCSTR3) CLASS(LARGE)
PATHOUT     STRNAME(IXCSTR4) CLASS(SMALL)
PATHIN      STRNAME(ISCSTR1,IXCSTR2,IXCSTR3,IXCSTR4)

DATA        TYPE(CFRM)
            PCOUPLE(SYS1.CFRMCDS.PRI)
            ACOUPLE(SYS1.CFRMCDS.ALT)
```

Figure 50. Example: specifying the CFRMPOL keyword in COUPLExx parmlib member

Chapter 9. Adding MVS systems to a sysplex

This topic describes how various sysplex configurations are set up. It includes the purpose and required parameters for each configuration and diagrams for many of the configurations. The following configurations are covered:

- [“XCF-local mode configuration” on page 201](#)
- [“Single-system sysplex configuration” on page 202](#)
- [“Multisystem sysplex configuration without an existing global resource serialization complex” on page 203](#)
- [“Multisystem sysplex configuration with an global resource serialization complex” on page 204](#)
- [“Multisystem sysplex configuration on one processor under z/VM” on page 208](#)
- [“Multisystem sysplex configuration on PR/SM” on page 209](#)
- [“Multisystem sysplex configuration with a coupling facility” on page 209](#)
- [“Multisystem sysplex configuration of greater than eight systems” on page 210.](#)

Because there are times when you need to remove a system from a sysplex, you can find information on how to do that in [“Removing a system from the sysplex” on page 210.](#)

XCF-local mode configuration

- **Purpose:** Use XCF-local mode when you:
 - Want to run a single system without sysplex services
 - Want to IPL the system to make changes, for example to change the names of the primary and alternate couple data sets specified in the COUPLExx parmlib member.
- **Considerations:** Multisystem applications can join XCF groups on an XCF-local system. Members of an XCF group can communicate with each other on the XCF-local system (via local message traffic). Note that on the IXCJOIN macro, LASTING=YES is not supported in XCF-local mode, which means that permanent status recording is not available to group members. Functions, such as WLM, that require a couple data set are not available because the sysplex couple data set required to support such couple data sets is not available.
- **Required System Parameters:** Use the following system parameters to initialize each MVS system:
 - COUPLE=00 (the default) specifies the default COUPLE00 parmlib member that indicates XCF-local mode for the initializing system. A couple data set is not specified for XCF-local mode.
 - PLEXCFG=XCFLOCAL indicates that the system is to be in XCF-local mode.

[Figure 51 on page 202](#) shows the multisystem environment with the MVS systems in XCF-local mode.

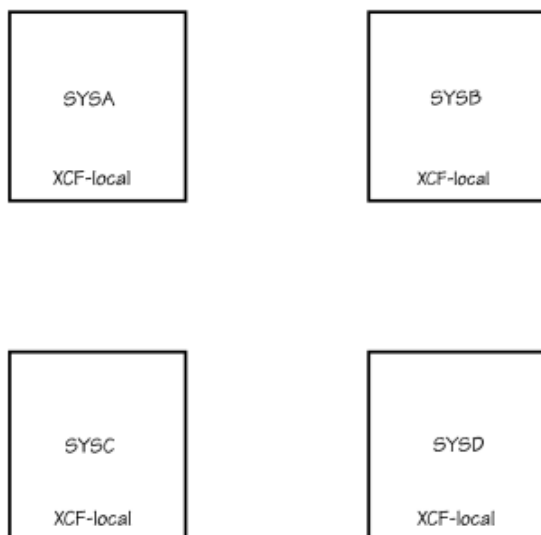


Figure 51. XCF-Local Mode Configuration

Single-system sysplex configuration

In single-system sysplex mode, you need a formatted primary sysplex couple data set on each system before you IPL the system.

- **Purpose:** Use single-system sysplex mode when you want XCF services on the system but do not need XCF or XES coupling services between systems. For example, you might want to implement a WLM policy on a single system, which requires a sysplex couple data set and a WLM couple data set, or test multisystem applications.
- **Considerations:** Multisystem applications can define XCF groups on the system and members of the group can communicate with their group members on the same system, but they cannot use signaling services to communicate with members on other systems. Note that, because a sysplex couple data set is in use, permanent status recording (LASTING=YES on the IXCJOIN macro) is available for group members on the single-system sysplex.

Consider this a temporary configuration until a multisystem sysplex is established.

- **Required System Parameters:** Use the following parameters to IPL the system:
 - COUPLE=xx specifies the COUPLExx parmlib member that describes the single-system sysplex for the system. COUPLExx must specify at least a sysplex name and a primary sysplex couple data set name. The sysplex couple data set must be unique for that system. COUPLExx must also specify the couple data set for any function, such as WLM, that you plan to use.
 - PLEXCFG=MONOPLEX indicates that the system is to be initialized as a single-system sysplex. Note that the system cannot migrate to a multisystem sysplex unless you specify PLEXCFG=MULTISYSTEM.

Figure 52 on page 203 shows a single-system sysplex (SYSA) connected to a sysplex couple data set and three independent systems. To migrate SYSB, SYSC, and SYSD into the sysplex, you must specify PLEXCFG=MULTISYSTEM for each system.

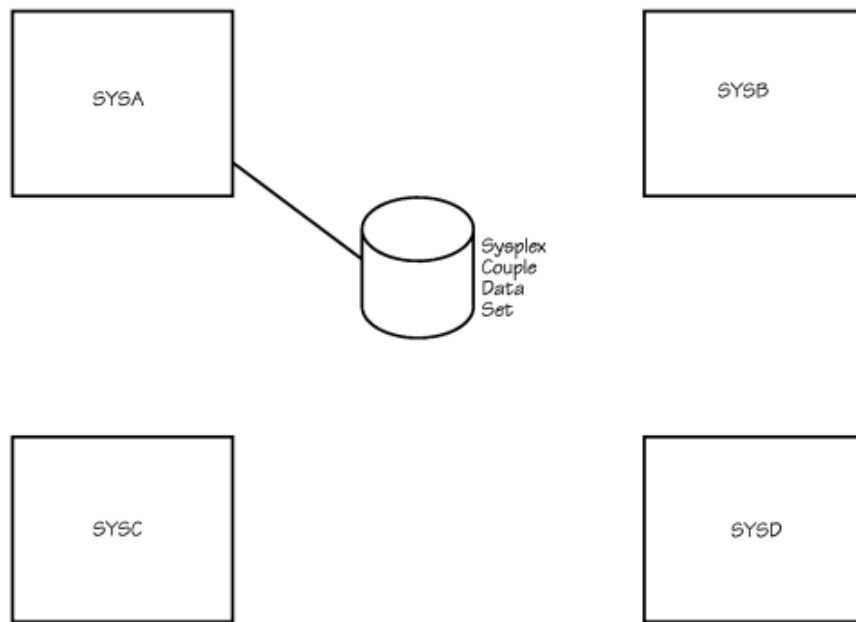


Figure 52. Configuration with a Single-System Sysplex and Three Independent Systems

Multisystem sysplex configuration without an existing global resource serialization complex

A sysplex requires a global resource serialization complex. The scope of the complex must encompass the scope of the sysplex. For information on creating a global resource serialization ring, see [z/OS MVS Planning: Global Resource Serialization](#).

This topic assumes that you do not have an existing global resource serialization complex and want to IPL four systems into a sysplex.

In multisystem sysplex mode, you need:

- A formatted primary sysplex couple data set shared by all systems in the sysplex.
- Signaling connectivity between all systems in the sysplex.
- The same Sysplex Timer shared by all systems in a sysplex that includes more than one CPC.

As the systems are IPLed and join the sysplex, a global resource serialization complex is created for the systems in the sysplex. Thus, the sysplex and complex match as you IPL the systems.

- **Purpose:** Use multisystem sysplex mode when you:
 - Want XCF group services on the systems and signaling services between systems in the sysplex.
 - Want to stop using the RESERVE macro to serialize global resources and start using the functions in a global resource serialization complex.
- **Required System Parameters:** Use the following system parameters to initialize each MVS system.
 - COUPLE=xx specifies the COUPLExx parmlib member that describes the sysplex environment for the initializing system. COUPLExx must specify at least the sysplex name, the primary couple data set name (which must be shared by all systems), and sufficient outbound and inbound signaling paths to obtain connectivity with all other systems in the sysplex. Defaults can be taken for other parameters.
 - PLEXCFG=MULTISYSTEM indicates that the system is to initialize as part of a multisystem sysplex.
 - GRS=TRYJOIN specifies that the system is to join an existing global resource serialization complex or start one.

- GRSCNF=00 (the default) specifies the GRSCNF00 parmlib member. GRSCNF00 contains global resource serialization parameters that can be used in the sysplex. You can specify GRSCNF=xx for a GRSCNFxx parmlib member if GRSCNF00 does not meet your installation's needs.
- GRSRNL=(xx,yy,...) specifies the GRSRNLxx parmlib member(s) that contain the resource name lists (RNLs) that you require for the global resource serialization complex to meet your installation needs. (The GRSRNL00 parmlib member contains IBM defaults.)
- CLOCK=xx specifies the CLOCKxx parmlib member that requests the Sysplex Timer to be used. MVS requires the Sysplex Timer to synchronize the coupling services.

As each of the MVS systems is IPLed, the sysplex and global resource serialization complex match, and consist of one, two, three, and then four systems. XCF coupling services and global resource serialization are available on the systems as they complete initialization.

Figure 53 on page 204 shows the multisystem environment after the four systems are IPLed.

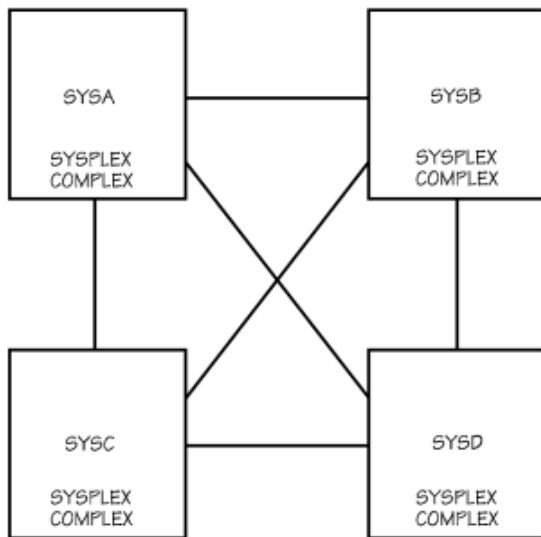


Figure 53. Multisystem Sysplex Configuration

After the four systems are IPLed, the multisystem sysplex configuration has:

- XCF coupling services available on all systems, including signaling services between all of the systems in the sysplex.
- The same primary sysplex couple data set shared by all systems in the sysplex (not shown).
- MVS component exploiters (such as global resource serialization) that exploit the XCF coupling services.
- Multisystem applications that can exploit the XCF coupling services in the sysplex.
- The same Sysplex Timer on all systems, which synchronizes time stamps for all systems (not shown).

Multisystem sysplex configuration with an global resource serialization complex

This topic assumes that you have an existing global resource serialization complex of four systems and want to initialize them into a sysplex that matches the complex.

Until all systems are IPLed and join the sysplex, a mixed complex exists; that is, one or more of the systems in the global resource serialization complex are not part of the sysplex.

In multisystem sysplex mode, you need:

- A formatted primary sysplex couple data set shared by all systems in the sysplex.
- Signaling connectivity between all systems in the sysplex

- The same Sysplex Timer for all systems in a sysplex that includes more than one CPC.
- **Purpose:** Use multisystem sysplex mode with an existing global resource serialization complex when you want XCF services on the initializing systems and signaling services between the systems in the sysplex.
- **Required System Parameters:** Use the following system parameters to initialize each MVS system:
 - COUPLE=xx specifies the COUPLExx parmlib member that describes the sysplex environment for the system. COUPLExx must specify at least the sysplex name, the primary couple data set name (which must be shared by all systems in the sysplex), and sufficient outbound and inbound paths to obtain full signaling connectivity with all other systems in the sysplex. Defaults can be taken for the other parameters.
 - PLEXCFG=MULTISYSTEM indicates that the system is to initialize as part of a multisystem sysplex.
 - GRS=TRYJOIN specifies that the system is to join an existing global resource serialization complex or start one.
 - GRSCNF=xx specifies the GRSCNFxx parmlib member for the existing global resource serialization complex. This includes the definitions of the necessary CTC links for the complex as systems are migrated into the sysplex. When the sysplex matches the complex, global resource serialization uses signaling services and no longer needs its own CTC links between systems.
 - GRSRNL=(xx,yy,...) specifies the GRSRNLxx parmlib member(s) that contain the resource name lists (RNLs) that you require for the complex to meet your installation needs.
 - CLOCK=xx specifies the CLOCKxx parmlib member that requests the Sysplex Timer to be used. MVS requires the Sysplex Timer to synchronize coupling services in the sysplex.

The following topics describe the multisystem environment as each of the four systems are initialized in order (SYSA, SYSB, SYSC, and then SYSD).

The first system IPL

Figure 54 on page 205 shows the multisystem environment after SYSA is initialized.

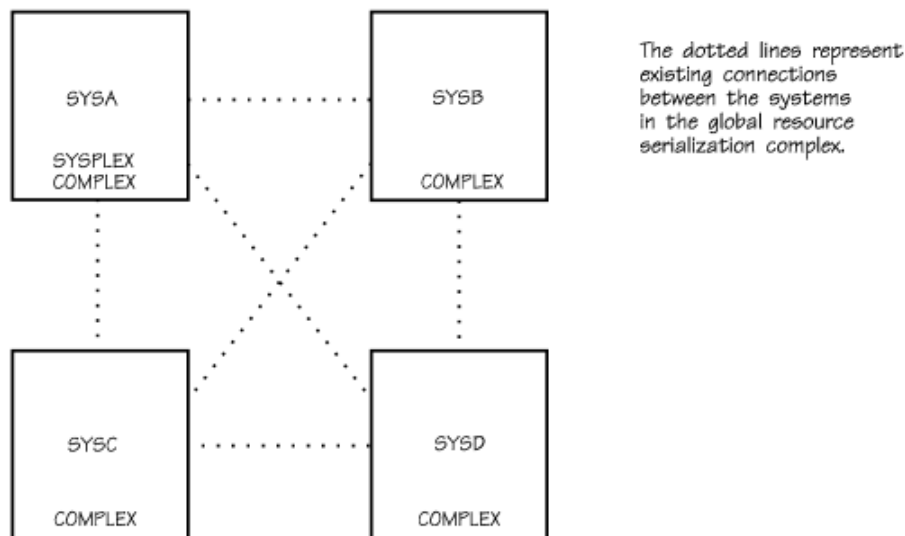


Figure 54. One System in the Multisystem Sysplex

When SYSA is initialized and joins the sysplex and the global resource serialization complex:

- XCF coupling services are available on SYSA, but signaling services are not available between any of the systems.
- A primary sysplex couple data set is used for SYSA (not shown).

- MVS component exploiters can exploit XCF coupling services on SYSA but must use non-XCF connections to communicate with other systems.
- Multisystem applications can exploit XCF coupling services on SYSA.

Consider this mixed complex a temporary configuration during migration until the sysplex matches the complex.

The second system IPL

Figure 55 on page 206 shows the multisystem environment after SYSB is initialized.

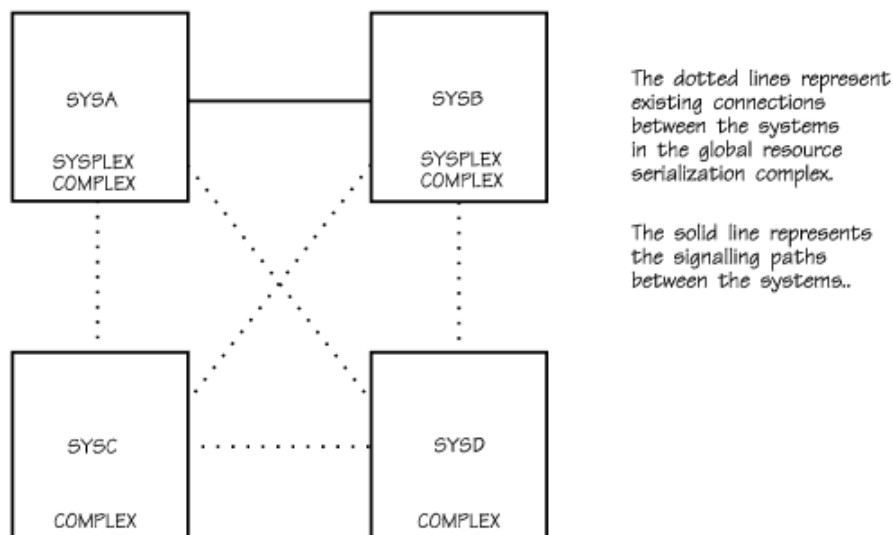


Figure 55. Two Systems in the Multisystem Sysplex

When SYSB is initialized and joins the sysplex and the global resource serialization complex:

- XCF coupling services are available on SYSA and SYSB, including signaling services between SYSA and SYSB.
- The same primary sysplex couple data set is shared by SYSA and SYSB (not shown).
- MVS component exploiters (such as global resource serialization) use signaling services between SYSA and SYSB but must use non-XCF connections to communicate with the other systems.
- Multisystem applications can exploit XCF coupling services and signaling services on SYSA and SYSB.
- SYSA and SYSB must share a Sysplex Timer if they are running on separate CPCs (not shown).

Consider this mixed complex a temporary configuration during migration until the sysplex matches the complex.

The third system IPL

Figure 56 on page 207 shows the multisystem environment after SYSC is initialized.

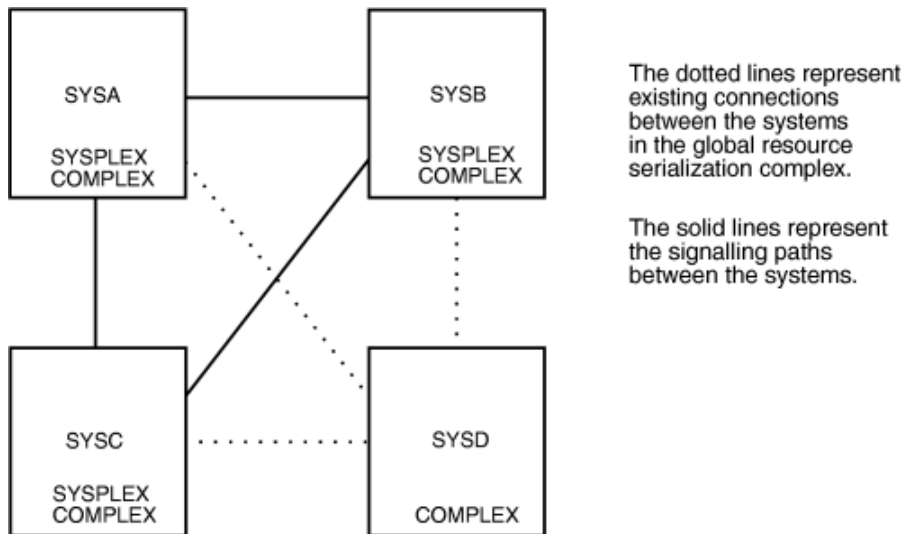


Figure 56. Three Systems in the Multisystem Sysplex

When SYSC is initialized and joins the sysplex and the global resource serialization complex:

- XCF coupling services are available on SYSA, SYSB, and SYSC, including signaling services between the three systems.
- The same primary sysplex couple data set (not shown) is shared by SYSA, SYSB, and SYSC.
- MVS component exploiters (such as global resource serialization) use signaling services between SYSA, SYSB, and SYSC but must use non-XCF connections to communicate with SYSD.
- Multisystem applications can exploit XCF coupling services and signaling services on SYSA, SYSB, and SYSC.
- SYSA, SYSB, and SYSC must share a Sysplex Timer if they are running on separate CPCs (not shown).

Consider this mixed complex a temporary configuration during migration until the sysplex matches the complex.

The fourth system IPL

Figure 57 on page 207 shows the multisystem environment after SYSD is initialized. The sysplex now matches the global resource serialization complex.

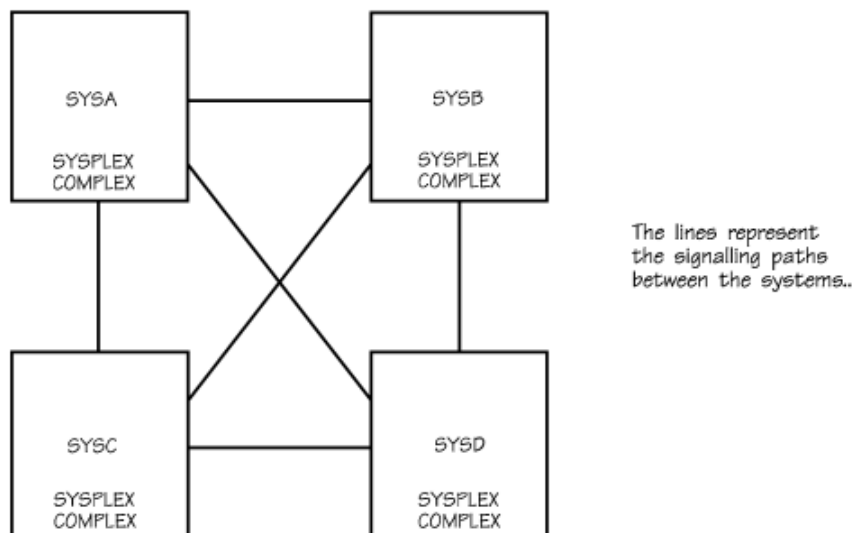


Figure 57. Four Systems in the Multisystem Sysplex

When SYSD is initialized and joins the sysplex, the sysplex matches the global resource serialization complex.

- XCF coupling services are available on all systems, including signaling services between all systems.
- The same primary sysplex couple data set is shared by all systems in the sysplex (not shown).
- MVS component exploiters (such as global resource serialization) use signaling services between all systems.
- Multisystem applications can use XCF coupling services and signaling services on all systems.
- The systems must share a Sysplex Timer if they are running on separate CPCs (not shown).
- **Advantages:** The following are some additional advantages when the multisystem sysplex matches the global resource serialization complex.
 - The CTC links assigned in the GRSCNFxx parmlib member for the global resource serialization complex are no longer needed because global resource serialization uses signaling services to communicate between systems.

See *z/OS MVS Planning: Global Resource Serialization* for the commands that you can use to vary the global resource serialization CTC links between SYSA and SYSB offline and then dynamically reassign them to XCF.

- There is reduced operator intervention when there is a ring disruption because global resource serialization can rebuild a global resource serialization ring without operator intervention.
- The resource name lists (RNLs) can be dynamically changed with the SET GRSRNL command while maintaining serialization integrity.

Multisystem sysplex configuration on one processor under z/VM

This topic describes a multisystem sysplex on one processor under z/VM. The z/VM system must be ESA capable. Note that an XCF-local mode or single-system sysplex can be established under z/VM similar to those shown in “XCF-local mode configuration” on page 201 and “Single-system sysplex configuration” on page 202. However, a sysplex running under z/VM cannot support a Sysplex Timer or a coupling facility.

- **Purpose:** Use multisystem sysplex mode under z/VM for the reasons listed in “Multisystem sysplex configuration without an existing global resource serialization complex” on page 203 and “Multisystem sysplex configuration with an global resource serialization complex” on page 204. In a multisystem sysplex on one processor under z/VM, a simulated Sysplex Timer must be used.

Each of the MVS systems in the sysplex must be under a single host z/VM on one processor.

- **Considerations:** A multisystem sysplex on one processor under z/VM should have APAR VM58844 installed. This APAR supports the CP SET command that you must issue for any DASD that contains any type of XCF couple data set to avoid integrity exposures.
 - SET WRKALLEG (for VM/ESA Release 1.2.2 and above)
 - SET SHARED (for VM/ESA releases prior to Release 1.2.2)

XCF will remove data sets that appear to have suffered an integrity exposure. If XCF removes both the primary and alternate data set for any type of XCF couple data, the function using that data set might cause one or more systems in the sysplex to enter a disabled wait state or might lose some of its function due to its inability to access its sysplex shared data.

- **Required System Parameters:** Use the following system parameters to initialize each MVS system:
 - CLOCK=xx specifies the CLOCKxx parmlib member that requests the same simulated Sysplex Timer (SIMETRID xx) on all systems.
 - Other system parameters for a multisystem sysplex as described in the preceding topics.

As each of the MVS systems are IPLed, the sysplex consists of one, two, three, and then four systems. XCF coupling services and global resource serialization are available on the systems as they complete initialization.

Multisystem sysplex configuration on PR/SM

This topic describes a multisystem sysplex on PR/SM. Note that an XCF-local mode or single-system sysplex can be established on PR/SM similar to those shown in [“XCF-local mode configuration” on page 201](#) and [“Single-system sysplex configuration” on page 202](#).

- **Purpose:** Use multisystem sysplex mode on PR/SM for the reasons listed in [“Multisystem sysplex configuration without an existing global resource serialization complex” on page 203](#) and [“Multisystem sysplex configuration with an global resource serialization complex” on page 204](#). A multisystem sysplex on one processor with PR/SM can use a Sysplex Timer or a simulated Sysplex Timer.
- **Required System Parameters:** Use the following system parameters to IPL each MVS system on one processor on PR/SM:
 - CLOCK=xx specifies the CLOCKxx parmlib member that requests either:
 - The same simulated Sysplex Timer (SIMETRID xx) on all systems, or
 - The same Sysplex Timer (ETRMODE YES) to be used on all systems.
 If the sysplex spans more than one processor, ETRMODE YES is required for all systems.
 - Other system parameters for a multisystem sysplex as described in the preceding topics.

As each of the MVS systems is IPLed, the sysplex consists of one, two, three, and then four systems. XCF coupling services and global resource serialization are available on the systems as they complete initialization.

See [“On PR/SM - Physical configuration of a sysplex on LPARs” on page 423](#) and [“Controlling availability and recovery through XCFPOLxx \(PR/SM only\)” on page 187](#) for additional information about PR/SM configurations.

Multisystem sysplex configuration with a coupling facility

This topic assumes that you have an existing multisystem sysplex and want to add a coupling facility to the configuration. In addition to the requirements for multisystem sysplex mode (see [“Multisystem sysplex configuration with an global resource serialization complex” on page 204](#)), you need:

- A sysplex couple data set formatted with the OS/390 format utility (IXCL1DSU) (not shown).
- A formatted primary CFRM couple data set shared by systems in the sysplex that use the coupling facility. The CFRM couple data set must contain at least one CFRM administrative policy.
- Connectivity between the coupling facility and the systems in the sysplex requiring its use.
- **Purpose:** Use a coupling facility to provide data sharing capability, the ability to isolate MVS systems through the SFM policy, or as an alternative to CTC connections for signaling.
- **Required System Parameters:** Use the following system parameter to IPL each system in the sysplex that is to use the coupling facility:
 - COUPLE=xx specifies the COUPLExx parmlib member that is updated to identify the primary CFRM couple data set.

Alternatively, you can use the SETXCF COUPLE,TYPE=CFRM,PCOUPLE command to bring the CFRM couple data set into use.

To start a policy in the CFRM data set, issue the SETXCF START,POLICY,TYPE=CFRM command, specifying the name of a CFRM policy that resides in the CFRM couple data set.

In [Figure 58 on page 210](#), SYSA and SYSB are connected to the coupling facility. (Required connections for the sysplex couple data set, Sysplex Timer, and signaling are not shown.)

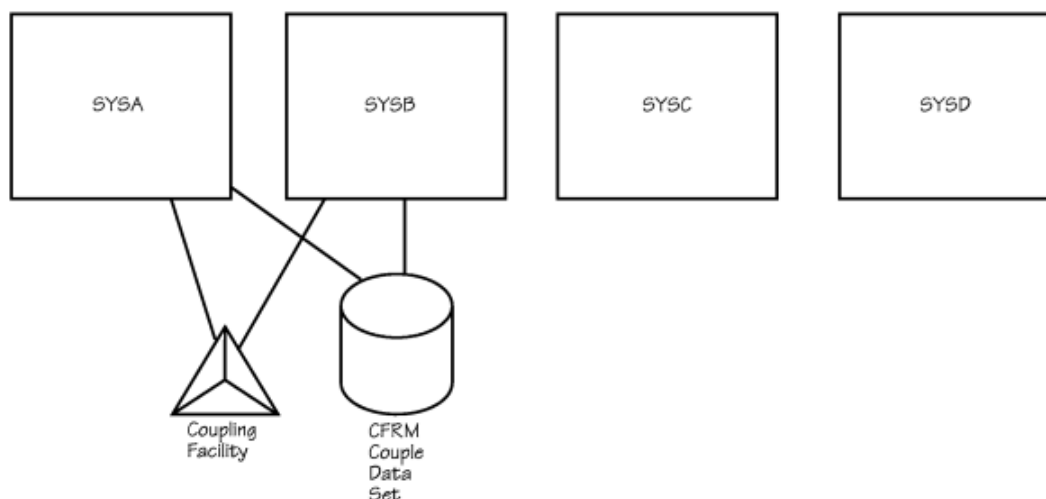


Figure 58. Multisystem Sysplex Configuration with a Coupling Facility

Multisystem sysplex configuration of greater than eight systems

This topic assumes that you have an existing multisystem sysplex and want to upgrade it to support more than eight systems. To operate a sysplex of more than eight systems, all systems must be at the OS/390 (or the MVS/ESA SP Version 5) level. This topic assumes that you are migrating your Version 4 systems to OS/390.

To operate a sysplex of more than eight systems, the primary sysplex couple data set must be formatted with the OS/390 format utility.

- **Migration Scenario:** Use the following migration scenario to upgrade your existing sysplex to support more than eight systems.
 1. One at a time, bring down the Version 4 systems in your sysplex and upgrade them to OS/390. Then, re-IPL each system into the sysplex. The system shares the same sysplex couple data set that it used prior to the upgrade. Continue in this way until each system in the sysplex is at the OS/390 level.
 2. In the meantime, use the OS/390 format utility to format a new sysplex couple data set to support more than eight systems. When all systems in the sysplex are at the OS/390 (or the MVS/ESA SP Version 5 level), introduce the new sysplex couple data set as the alternate, using the SETXCF COUPLE,ACOUPLE command. You can then use the SETXCF COUPLE,PSWITCH command to make the alternate sysplex couple data set the primary.

At this point, the sysplex can make use of the ARM, CFRM, SFM, and WLM couple data sets.

3. New OS/390 systems can now join the sysplex, up to the limit for which the sysplex couple data set is formatted. The limit is specified in the format utility on the MAXSYSTEM keyword of the DEFINEDS statement.

Note the following restrictions:

- Once the sysplex couple data set formatted with the OS/390 format utility becomes the primary sysplex couple data set, you cannot use as an alternate sysplex couple data set one that has been formatted with the SP Version 4 format utility.
- You can no longer introduce an MVS/ESA SP Version 4 system to the sysplex.

Removing a system from the sysplex

Removing a system from the sysplex means that:

- All XCF group members in the sysplex know that the system is being removed, so they can perform any necessary cleanup processing.
- All I/O to sysplex-shared resources is completed, to ensure data integrity.
- All signaling paths are configured to a desired state (retained or not).

The following are the basic steps for removing a system from a sysplex. For detailed instructions for the necessary steps, see “Removing a z/OS image from a Sysplex” at [Parallel Sysplex \(www.ibm.com/systems/z/advantages/pso/removing.html\)](http://www.ibm.com/systems/z/advantages/pso/removing.html).

1. Use the VARY XCF command to remove a system from the sysplex. You can remove the system temporarily or permanently.

To temporarily remove a system from the sysplex, issue:

```
VARY XCF,system-name,OFFLINE,RETAIN=YES
```

With RETAIN=YES (the default), MVS on each remaining system in the sysplex retains the definition of the devices for the signaling paths that connected to the removed system. Therefore, the removed system can be re-IPLed into the sysplex or another system can be added in its place, and MVS automatically starts the signaling paths. Note that the last system removed from the sysplex, remains defined to the sysplex couple data set.

To permanently remove a system from the sysplex, issue:

```
VARY XCF,system-name,OFFLINE,RETAIN=NO
```

With RETAIN=NO, MVS on each remaining system stops the signaling paths connected to the removed system and discards their definition. Therefore, you cannot re-IPL the system into the sysplex.

2. Reply to message IXC371D, which requests confirmation of the VARY XCF command.

Message IXC101I then informs you that the system is being removed.

3. When the target system is in a wait state (issues message IXC220W with a wait state code 0A2), issue a system reset. The reset must be done after the wait state, to ensure the integrity of I/O to sysplex shared I/O resources.

4. Reply DOWN to message IXC102A to continue removing the system from the sysplex.

After you reply, when the removal of the system is complete, message IXC105I is issued.

Note: Steps 3 and 4 can be automated using the SFM policy. See [“Controlling system availability and recovery through the SFM Policy”](#) on page 180.

Lock structure considerations

If a system partitioned out of the sysplex was managing one or more locks for one or more coupling facility lock structures, then XES must transfer that management to remaining sysplex peer systems during post-partition cleanup. During this transition, XES will quiesce all activity to the associated lock structures in order to preserve data integrity. This delay may impact subsystems that use these coupling facility lock structures. The ISGLOCK structure used by a Global Resource Serialization Star complex is an example of this, temporarily preventing the processing of SCOPE=SYSTEMS resources. Other coupling facility lock structure users include IRLM/DB2, VSAM RLS, and IRLM/IMS.

Multisystem application considerations

When you issue the VARY XCF,system-name,OFFLINE command, MVS notifies all groups, through their member group routines, that the system is to be removed from the sysplex. Multisystem applications that are not members of a group can also be notified of a system being removed from the sysplex through ENF event code 35. It is an application program's responsibility to ensure that its remaining members on the other systems in the sysplex can continue processing when a system is removed from the sysplex.

If the system is re-IPLed into the sysplex, MVS notifies all groups, through their member group routines, that the system is added to the sysplex. Multisystem applications that are not members of an XCF group can be notified of a system joining the sysplex through ENF event code 35.

Rejoining a sysplex

There are instances when a system remains defined in the couple data set as “active” even though measures have been taken to remove the system from the sysplex. For example, the last system removed from a sysplex with a VARY command remains defined in the couple data set as an “active” system. Before that system can rejoin the sysplex, it must be determined whether the sysplex should be reinitialized (implying the cleanup of residual system information). The following applies:

- In native MVS mode, when the system attempts to rejoin the sysplex, XCF is able to determine that the system is a new instance of the “active” system that is defined in the couple data set. As such, it can reinitialize the couple data set without requiring operator intervention.
- In a VM guest environment, with OW31481 or OW32876 installed, when the VM guest system attempts to rejoin the sysplex, XCF determines whether the system is a new instance of the “active” system defined in the couple data set by verifying that:
 - Both systems are VM guests.
 - Both systems have the same system name.
 - The system status of the system identified in the couple data set has not been updated in a reasonable time frame prior to the attempt to rejoin the sysplex.

If these conditions exist, XCF will initialize the couple data set without requiring operator intervention.

When the system attempting to rejoin the sysplex has a different system name from the “active” system defined in the couple data set and the status of the “active” system has not been updated in a reasonable time frame, the operator is prompted to specify whether the new system should join or initialize the sysplex.

Chapter 10. Planning for system logger applications

This section covers planning steps for any system logger applications. It includes the following topics:

- [“What is system logger?” on page 213](#)
- [“The system logger configuration” on page 215](#)
- [“Finding information for system logger applications” on page 239](#)
 - [“Finding information for CICS log manager” on page 239](#)
 - [“Finding information for OPERLOG log stream” on page 240](#)
 - [“Finding information for logrec log stream” on page 241](#)
 - [“Finding information for APPC/MVS” on page 242](#)
 - [“Finding information for IMS common queue server log manager” on page 242](#)
 - [“Finding information for resource recovery services” on page 243](#)
 - [“Finding information for system management facilities \(SMF\)” on page 243](#)
- [“Preparing to use system logger applications” on page 243](#)
 - [“Understand the requirements for system logger” on page 244](#)
 - [“Plan the system logger configuration” on page 247](#)
 - [“Determine the size of each coupling facility structure” on page 255](#)
 - [“Develop a naming convention for system logger resources” on page 258](#)
 - [“Plan DASD space for system logger” on page 260](#)
 - [“Managing log data: How much? For how long?” on page 272](#)
 - [“Define authorization to system logger resources” on page 273](#)
 - [“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276](#)
 - [“Add information about log streams and coupling facility structures to the LOGR policy” on page 280](#)
 - [“Deleting log streams from the LOGR, LOGRY or LOGRZ policy” on page 289](#)
 - [“Define the coupling facility structures attributes in the CFRM function couple data set” on page 289](#)
 - [“Activate the LOGR subsystem” on page 291](#)
- [“Deleting log data and log data sets” on page 291](#)
- [“Upgrading an existing log stream configuration” on page 293](#)
 - [“Upgrading a log stream from DASD-only to coupling facility” on page 295](#)
 - [“Updating a log stream to use a different coupling facility structure” on page 296](#)
- [“System logger recovery” on page 300](#)

What is system logger?

System logger is an MVS component that allows an application to log data from a sysplex. You can log data from one system or from multiple systems across the sysplex. A system logger application can be supplied by:

- IBM. For example, CICS log manager and the operations log stream (OPERLOG) are IBM-supplied system logger applications.
- Independent software vendors.
- Your installation.

A system logger application can write log data into a **log stream**, which is simply a collection of data. Data in a log stream spans two kinds of storage:

- **Interim storage**, where data can be accessed quickly without incurring DASD I/O.
- **DASD log data set storage**, where data is hardened for longer term access. When the interim storage medium for a log stream reaches a user-defined threshold, the log data is off-loaded to DASD log data sets.

There are two types of log streams: coupling facility log streams and DASD-only log streams. The main difference between the two types of log streams is the storage medium system logger uses to hold interim log data:

- In a coupling facility log stream, interim storage for log data is in coupling facility list structures. See [“Coupling facility log stream” on page 214](#).
- In a DASD-only log stream, interim storage for log data is contained in local storage buffers on the system. **Local storage buffers** are high virtual private areas associated with the system logger address space, IXGLOGR. See [“DASD-only log stream” on page 215](#).

Your installation can use just coupling facility log streams, just DASD-only log streams, or a combination of both types of log streams. The requirements and preparation steps for the two types of log streams are somewhat different; see [“Preparing to use system logger applications” on page 243](#).

Coupling facility log stream

Figure 59 on page 214 shows how a coupling facility log stream spans two levels of storage: the coupling facility for interim storage and DASD log data sets for more permanent storage. When the coupling facility space for the log stream fills, the data is off-loaded to DASD log data sets. A coupling facility log stream can contain data from multiple systems, allowing a system logger application to merge data from systems across the sysplex.

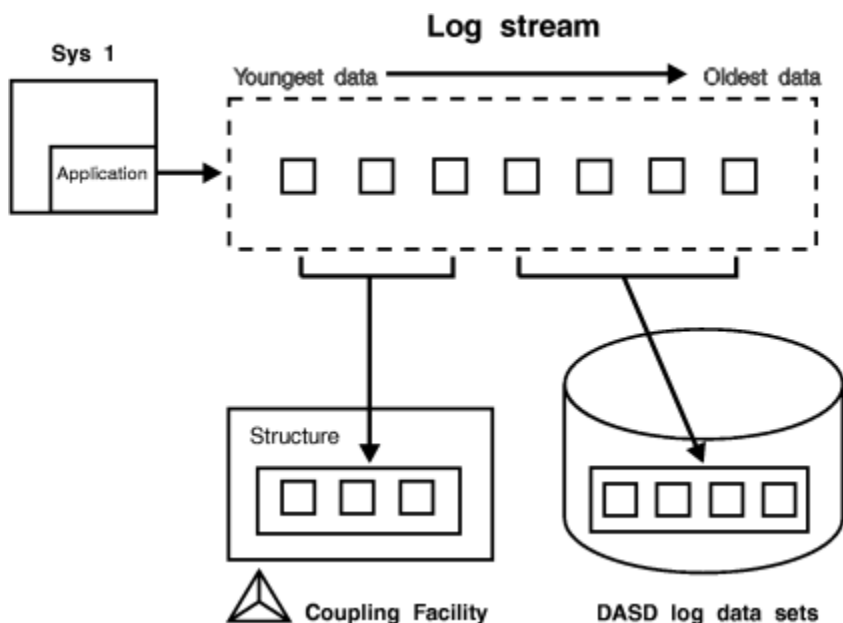


Figure 59. Log Stream Data on the Coupling Facility and DASD

When a system logger application writes a log block to a coupling facility log stream, system logger writes it first to a coupling facility list structure. System logger requires that a coupling facility list structure be associated with each log stream. When the coupling facility structure space allocated for the log stream reaches the installation-defined threshold, system logger moves (**offloads**) the log blocks from the coupling facility structure to VSAM linear DASD data sets, so that the coupling facility space for the log stream can be used to hold new log blocks. From a user's point of view, the actual location of the log data in the log stream is transparent.

DASD-only log stream

Figure 60 on page 215 shows a DASD-only log stream spanning two levels of storage: local storage buffers for interim storage, which is then off-loaded to DASD log data sets for more permanent storage.

A DASD-only log stream has a single-system scope; only one system at a time can connect to DASD-only log stream. Multiple applications from the same system can, however, simultaneously connect to a DASD-only log stream.

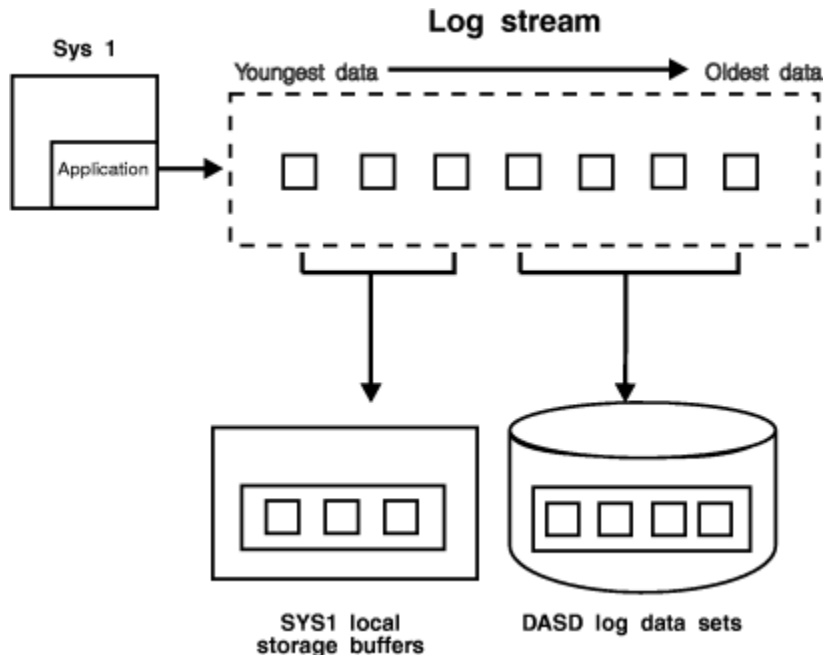


Figure 60. Log Stream Data in Local Storage Buffers and DASD Log Data Sets

When a system logger application writes a log block to a DASD-only log stream, system logger writes it first to the local storage buffers for the system and duplexes it to a DASD staging data set associated with the log stream. When the staging data set space allocated for the log stream reaches the installation-defined threshold, system logger off-loads the log blocks from local storage buffers to VSAM linear DASD data sets. From a user's point of view, the actual location of the log data in the log stream is transparent.

Both DASD-only log streams and coupling facility log streams can have data in multiple offload data sets. System Logger automatically allocates new offload data sets for log streams.

The system logger configuration

The system logger configuration you use depends on whether or not you use a coupling facility.

Coupling Facility Log Stream Configuration: Figure 61 on page 216 shows all the parts involved when a system logger application writes to a coupling facility log stream. In this example, a system logger application runs on two systems in a sysplex. Both instances of the application write data to the same log stream, TRANSLOG. Each system contains a system logger address space. A system logger application uses system logger services to access the system logger capabilities.

When a system logger application writes data to a coupling facility log stream, system logger writes the data to a coupling facility list structure associated with the log stream. Then, when the coupling facility structure fills with data, system logger off-loads the data to DASD log data sets.

You can optionally elect to have coupling facility data duplexed to DASD staging data sets for a coupling facility log stream.

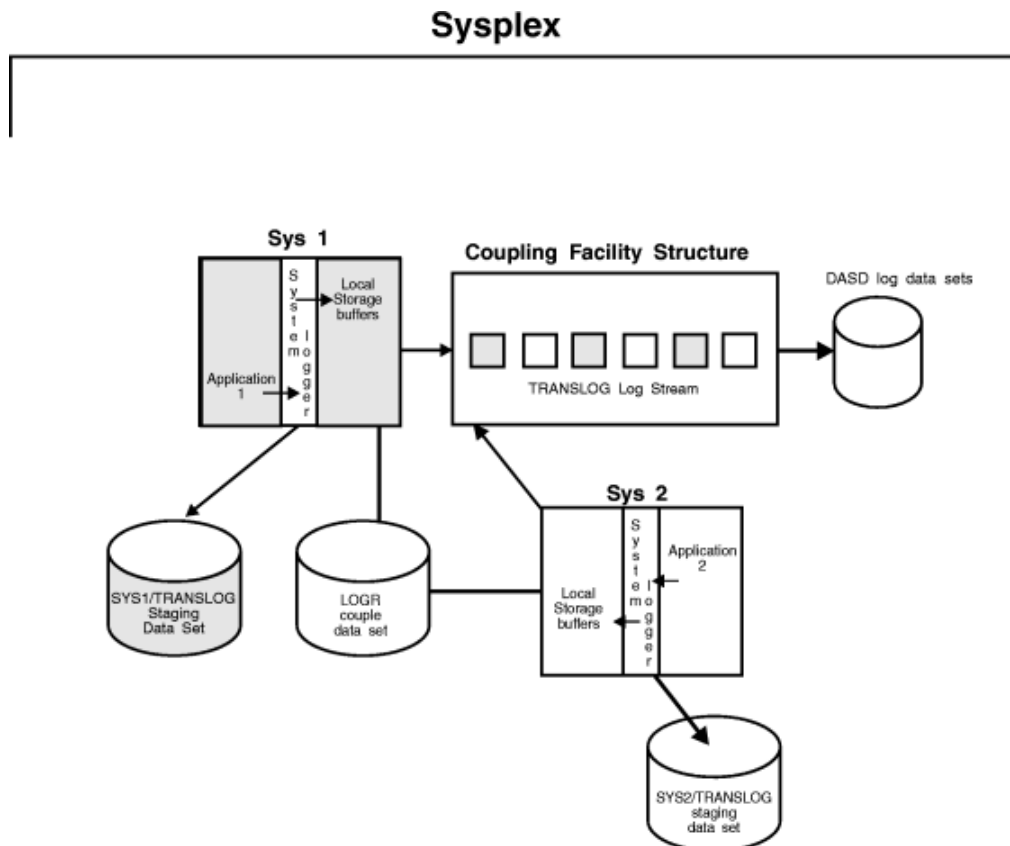


Figure 61. A Complete Coupling Facility Log Stream Configuration

DASD-Only Log Stream Configuration: Figure 62 on page 216 shows all the parts involved when a system logger application writes to a DASD-only log stream. System logger writes the data to the local storage buffers on the system, duplexing it at the same time to the DASD staging data sets associated with the log stream. Then, when the staging data set fills with data, system logger off-loads the data to DASD log data sets. Note that where duplexing to DASD staging data sets is an option for a coupling facility log stream, it is a required automatic part of a DASD-only log stream.

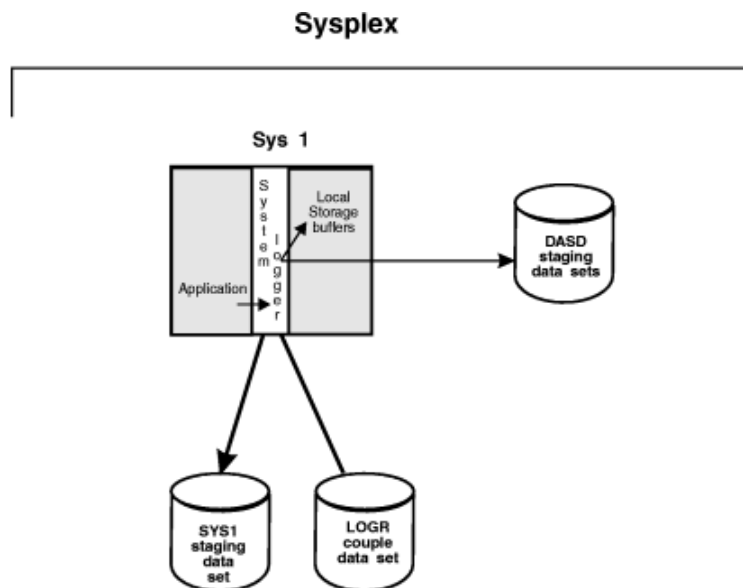


Figure 62. A DASD-Only Configuration

The system logger component

The system logger component resides in its own address space, IXGLOGR, on each system in a sysplex. Some of the component processing will differ, depending on whether a given log stream is a coupling facility log stream or a DASD-only log stream. The system logger component does the following:

- Provides a set of system services that allows a system logger **application** to use the system logger component. See *z/OS MVS Programming: Assembler Services Guide* and *z/OS MVS Programming: Assembler Services Reference IAR-XCT* for writing a system logger application.
- Maintains information in the LOGR, LOGRY and LOGRZ policies about the current use of log streams and if used, coupling facility list structures.
- For coupling facility log streams, system logger interacts with cross-system extended services (XES) to connect to and use the coupling facility for system logger applications.
- Obtains local storage buffer space. For a coupling facility log stream, local storage buffers can be used for duplexing log data. For a DASD-only log stream, local storage buffers are used as interim storage for log data before it is off-loaded to DASD log data sets.
- Off-loads data to DASD log data sets as follows:

For coupling facility log streams, system logger off-loads log data from the coupling facility to DASD log data sets as the coupling facility structure space associated with the log stream reaches the installation-defined thresholds.

For DASD-only log streams, system logger off-loads log data from the local storage buffers to DASD log data sets as the DASD staging data set space reaches the installation-defined thresholds.

- Automatically allocates new DASD log data sets for log streams.
- Maintains a backup copy of (duplexes) log data that is in interim storage for recovery. Log data in interim storage is vulnerable to loss due to system or sysplex failure because it has not yet been hardened to DASD log data sets. System logger duplexes interim storage log data for both coupling facility and DASD-only log streams.
- Produces SMF record type 88 for system logger accounting on a single system. Record type 88 focuses on the usage of interim storage (coupling facility or local storage buffers) and log stream data for a system in the sysplex. Using the record can help an installation avoid the STRUCTURE FULL or STAGING DATA SET FULL exceptions, and perform other tuning and/or capacity planning analysis.

See *z/OS MVS System Management Facilities (SMF)* for more record type 88 and system logger accounting. Sample program IXGRPT1 in SYS1.SAMPLIB shows an example of producing a report from SMF record type 88.

- Ensures that:
 - When the last connection from a system disconnects from the log stream, all log data written by that system to the log stream is off-loaded to DASD log data sets.
System logger also deletes any staging data sets in use for a system at this time.
 - When the last connection to a coupling facility log stream in the sysplex disconnects, all coupling facility log data is off-loaded to DASD log data sets and the coupling facility space is returned to XES for reallocation.
- Provides recovery support in the event of application, system, sysplex, or coupling facility structure failure for coupling facility log streams. (See “System logger recovery” on page 300 for information about recovery for DASD-only log streams.)

Indicating the IXGLOGR address space is not to start on a system

The LOGR subsystem specification generally allows you to read log stream data without having to rewrite your existing applications. See “Activate the LOGR subsystem” on page 291 for more this topic.

However, there are sysplex configurations where it would be appropriate to not allow the IXGLOGR address space to start on a system. For example, in certain Geographically Dispersed Parallel Sysplex

(GDPS) configurations, the controlling, or k-, system may need to have System Logger not active to avoid any log stream activity from occurring on that system.

If you need to have the IXGLOGR address space not be started on a system, specify the following:

- In an IEFSSNxx parmlib member and use the member during the system IPL:

```
SUBSYS SUBNAME(LOGR) INITRTN(IXGSSINT) INITPARM(IXGLOGR=NOSTART)
```

Since IEFSSNxx parmlib members cannot contain the new and old format definitions in the same IEFSSNxx parmlib member, you might need to define a new parmlib member for the LOGR subsystem specification and concatenate it with other IEFSSNxx parmlib members.

- In a COMMNDxx parameter member used during the IPL:

```
COM= 'SETSSI ADD, SUBNAME=LOGR, INITRTN=IXGSSINT, INITPARM= ' ' IXGLOGR=NOSTART ' ' '
```

If a SETSSI command is issued after the first initialization of the IXGLOGR address space then the INITPARM option is ignored and will have no effect on the IXGLOGR address space.

Note: The above format for specifying the LOGR subsystem options is required, but the installation can choose any valid subsystem name using the SUBNAME keyword (it does not have to be LOGR). Make certain, however, that the subsystem name specified in the SUBNAME parameter matches the corresponding name in the SUBSYS option in any JCL or dynamic allocation intended to use the system logger subsystem functions.

The IXGLOGR=NOSTART specification in the INITPARM keyword will result in the following system action:

1. The LOGR or named subsystem (*ssname*) functional routines will be established.

However, any JCL DD SUBSYS=(LOGR,exit,...) or dynamic allocation equivalents will likely result in the subsystem exit routine encountering a return code 8, reason code X'0814' condition (see IxgRsnCodeNotAvailForIPL in IXGCON macro).

For example, if the logger subsystem default exit IXGSEXIT was requested in this environment, then Logger message IXG511I would be issued indicating the IXGCONN service failed with the return and reason codes listed above.

Errors returned by any Logger subsystem exit routine will cause Logger to issue message IXG504I indicating the exit error condition.

2. The IXGLOGR address space will not complete its initialization at the end of Master Scheduler Initialization (MSI), and the address space will terminate.

Attempts to start the IXGLOGR address space (re: IXGLOGRS procedure) will not complete successfully as Logger will not allow the IXGLOGR address space to start because of the IXGLOGR=NOSTART specification.

3. Message IXG053I is issued indicating the IXGLOGR address space is not available until another IPL is done for this system.
4. Any System Logger API (IXGxxxxx) service request made on this system will result in the requestor receiving a return code 8, reason code X'0814' (IxgRsnCodeNotAvailForIPL) condition.

The LOGR couple data set (LOGR policy)

The system logger component manages log streams based on the policy information that installations place in the LOGR couple data set. The LOGR couple data set must:

- Be accessible by all the systems in the sysplex, unless explicitly intended otherwise.

See [“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276](#) and [“The system logger single-system scope couple data sets \(LOGRY and LOGRZ\) policies” on page 219](#).

- Be formatted using the IXCL1DSU utility.

- Contain policy information, which is defined using the IXCMIAPU utility or the IXGINVNT service. (See [z/OS MVS Programming: Assembler Services Guide](#) for information about the IXGINVNT service.)

You can have only one LOGR policy for the sysplex, even if you have primary and alternate LOGR couple data sets. System logger will only use the policy information contained in the primary LOGR couple data set. Because there is only one LOGR policy allowed for a sysplex, you cannot specify the DEFINE POLICY NAME parameter in the LOGR policy, nor can you issue the SETXCF START,POLICY command to activate a policy. Updates made to the LOGR policy are made in the primary LOGR couple data set. If you switch to your alternate LOGR couple data set (making it now the primary LOGR couple data set), XCF had already copied and maintained the contents of the LOGR policy from the old primary couple data set into the new one.

The LOGR policy includes the following:

- policy information:
 - log stream definitions
 - coupling facility list structure definitions, if applicable
 - log stream directory records (DSEXTENTS)
- data containing the current state of log streams, for example:
 - whether a log stream is currently connected to the coupling facility structure
 - the duplex state on a system connection basis
 - the directory of log stream offload data sets that house log data moved from interim to secondary log stream storage

Formatting a LOGR couple data set and defining the LOGR policy are steps an installation must take to prepare for using a system logger application. See [“Preparing to use system logger applications” on page 243](#).

The system logger single-system scope couple data sets (LOGRY and LOGRZ) policies

In addition to the sysplex scope LOGR policy, system logger supports up to two single-system scope policies (LOGRY and LOGRZ) in a sysplex. The LOGRY and LOGRZ policy information is similar to the LOGR policy information, except for the single-system scope policies no structures can be defined and the log streams must be DASD-only type. The log streams in the LOGRY and LOGRZ policies are managed as separate resources from the log stream resources defined in the sysplex scope LOGR policy and the other single-system scope policy. Only one system in the sysplex can be using the LOGRY policy and only one other system can be using the LOGRZ policy as their respective active CDS data type. If a system is using either the LOGRY or LOGRZ single-system scope policy, then that system cannot be using the sysplex scope LOGR policy.

When using a single-system scope CDS data type in your sysplex, determine if there is any need to understand the system logger configuration and distinguish whether a logger resource or activity is of sysplex scope or system scope. An anticipated sysplex scope could be when there is a singularly named log stream for the sysplex. For example, the log stream name SYSplex.OPERLOG is the z/OS Consoles system messages common repository for any system in the sysplex with the policy specification to use OPERLOG. Ensure your installation procedures are appropriate for managing any distinct sysplex scope and single-system scope log stream resources.

The LOGRY or LOGRZ couple data set must:

- Be accessible by the one system in the sysplex that is intended to make use of the specific CDS type. See [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex” on page 279](#).
- Be formatted using the IXCL1DSU utility.
- Contain policy information which is defined using the IXCMIAPU utility or the IXGINVNT service. For information about the IXGINVNT service, see [z/OS MVS Programming: Assembler Services Guide](#).

You can have only one of each for the LOGRY policy and LOGRZ policy for the sysplex, even if you have primary and alternate couple data sets for each LOGRY and LOGRZ policy. System logger will only use the policy information contained in the active primary single-system couple data set. Because there is only one LOGRY policy and only one LOGRZ policy allowed for a sysplex, you cannot specify the DEFINE POLICY NAME parameter in the LOGRY or LOGRZ policy, nor can you issue the SETXCF START,POLICY command to activate a policy.

Updates made to the LOGRY policy are made in the primary LOGRY couple data set and updates made to the LOGRZ policy are made in the primary LOGRZ couple data set. If you switch to your alternate LOGRY or LOGRZ couple data set (making it now the primary LOGRY or LOGRZ couple data set, respectively), XCF will have already copied and maintained the contents of the LOGRY and LOGRZ policy from its old primary couple data set into the new one.

Both the LOGRY policy and the LOGRZ policy include the following:

- policy information:
 - log stream definitions (DASD-only types)
 - log stream directory records (DSEXTENTS)
- data containing the current state of log streams, for example:
 - whether a log stream is currently connected on a system
 - the duplex state
- the directory of log stream offload data sets that house log data moved from interim to secondary log stream storage.

Formatting a LOGRY or LOGRZ couple data set and defining the LOGRY and LOGRZ policies are steps an installation must take to prepare for using a system logger application. See [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex”](#) on page 279 and [“Prevent a z/OS image from accessing sysplex scope LOGR couple data sets”](#) on page 277 for more details on formatting a LOGRY and LOGRZ couple data set. Also see [“The system logger single-system scope couple data sets \(LOGRY and LOGRZ\) policies”](#) on page 219 for details on specifying system logger policy information in a single-system scope couple data set.

For details on what information can be viewed and acted upon for these single-system scope couple data sets, refer to the Display LOGGER and SETLOGR commands in [z/OS MVS System Commands](#).

Log data on the coupling facility

For a coupling facility log stream, system logger requires that a coupling facility list structure be assigned for each log stream as interim storage. When an application writes data to a coupling facility log stream, it is first written to the coupling facility list structure for that log stream. System logger manages connections to the coupling facility, but the installation must define coupling facility structures dedicated to log streams in the LOGR policy as part of the planning process for system logger applications. More than one log stream can share space in a single structure.

System logger will dynamically manage the entry-to-element ratio for log stream coupling facility structures. The installation defines an initial average buffer size for a structure on the AVGBUFSIZE parameter in the LOGR policy using the IXGINVNT service or the IXCMIAPU utility. System logger uses this value to determine the initial entry-to-element ratio for the structure. After that, system logger will dynamically manage the entry-to-element ratio for the structure based on actual structure usage. The most current entry-to-element ratio is used on any subsequent allocations of the structure.

Log data on DASD log data sets

System logger stores log data on DASD in VSAM linear data sets when the log data in interim storage either reaches its installation defined threshold or fills. The VSAM linear data sets used for a log stream are called the log stream offload data sets. Both a DASD-only log stream and a coupling-facility log stream can have data in multiple offload data sets on DASD. System logger automatically allocates new ones for the log stream. During this offload data sets switch activity, System logger increments the sequence

number (suffix qualifier) in the log stream data set name as new data sets are added for a particular log stream. See [“Naming conventions for the log stream and DASD data sets” on page 258](#) for information on using SMS.

IBM suggests that you use System Managed Storage (SMS) to manage DASD log data sets. See [“Specifying SMS data set characteristics for DASD data sets” on page 281](#) for more information on using SMS.

System logger can use DFHSM or another product to migrate DASD log data sets. See [“Log stream data set migration and recall considerations” on page 268](#) for considerations on migrating and recalling log stream data sets.

Duplexing log data

Every time a system logger application writes a log block to a log stream, system logger automatically ensures there is a duplicate copy of the data as insurance against data loss due to coupling facility, system, or system logger failure. System logger keeps a duplex copy of data in log stream interim storage only. The duplicate copy is kept until the data is off-loaded from interim storage to DASD log data sets.

The way system logger duplexes log data depends on whether the log stream is a coupling facility or a DASD-only log stream.

Duplexing log data on a coupling facility log stream

System logger allows the installation to choose between different methods of duplexing coupling facility log data for a structure-based log stream:

- Maintain a copy of coupling facility resident log data in local storage buffers on each system.

This option is the default, and occurs automatically if you do not specifically request duplexing on staging data sets in the log stream definition in the LOGR policy.

- Maintain a copy of coupling facility resident log data in DASD **staging data sets**, one per log stream on a system.

This option provides hardening of the data on failure independent, persistent media.

- Maintain a copy of coupling facility resident log data in a structure eligible for system-managed duplexing rebuild.

This option allows the coupling facility structure to be duplexed automatically, meaning that there will be two instances of the structure, one being the copy of the other. By using system-managed duplexing, a copy of the structure log data can be on a coupling facility that is separate from the primary instance of the structure. This provides the capability of having a backup of the structure log data at an “off-site” location (within the bounds of the CF-to-CF links).

The duplexing method for a coupling facility log stream is established on the log stream definition in the LOGR policy along with the CFRM policy structure definitions. See [“Selecting a method of duplexing coupling facility log data” on page 283](#) for details on how to specify the duplexing options that will be used to backup the log stream data in the structure.

As a default, system logger will provide its own log data duplexing. However, the logger duplexing can be used in conjunction with system-managed duplexing or it can be replaced by system-managed duplexing.

You can request the duplexing options for a coupling facility log stream in the log stream definition in the LOGR policy.

You can request duplexing to staging data sets either conditionally, so that a connection will have a staging data set when it contains a single point of failure, or unconditionally, so that every connection to a log stream duplexes to staging data sets. This means that the number of staging data sets active at any given time will depend on the status of each connection and the duplexing method selected for each log stream.

The log data duplexing to the staging data set can be done in a synchronous manner or asynchronous manner. Normally the log data will be duplexed synchronously, meaning that after it is written to the

primary log stream storage, the log data is then written to the staging data set before Logger returning to the log writer with the results of the write request. However, when specifically requested, the log data can be duplexed asynchronously, meaning that after it is written to the primary log stream storage, the log data is then buffered in an I/O buffer but may be written to the staging data set after Logger returns to the log writer with the results of the write request.

If you request duplexing using staging data sets for a coupling facility log stream in the LOGR policy, system logger uses VSAM linear DASD staging data sets to hold a back-up copy of coupling facility log data for log streams that specify it.

IBM suggests that you plan for DASD staging data sets even if you do not request duplexing to staging data sets. If the primary LOGR couple data set for the sysplex is at OS/390 Release 3 or above, system logger will automatically allocate a staging data set for a log stream under specific error conditions where the log data exists only in a volatile configuration. System logger does this to help safeguard log data against data loss.

Duplexing for a DASD-only log stream

For a DASD-only log stream, system logger automatically duplexes log data from the system's local storage buffers to DASD staging data sets. System logger uses VSAM linear DASD data sets for staging data sets. Note that the staging data set naming convention is different for DASD-only and coupling facility log streams.

Offloading log data from interim storage by freeing and/or moving it to DASD

As one or more connectors write data to a log stream, the log stream's interim storage medium begins to fill up, eventually reaching or exceeding its high threshold. If that interim storage medium fills completely, the log stream is not able to accept new IXGWRITE requests.

Log stream offload processing is the mechanism by which system logger deletes and/or moves log data from interim storage to offload data sets. This frees up space in the interim storage medium and ensures that the log stream can continue to accept new IXGWRITE requests.

You specify (in the LOGR, LOGRY or LOGRZ policy) high and low thresholds for each log stream to define when system logger begins and ends the process of moving, or offloading, log data to DASD log data sets. It is important for you to understand the way system logger offloading works so that:

- You can set the high and low thresholds that control offloading.
- You understand when, why, and how often offloading will occur for your installation.
- You understand when log data is physically deleted from the log stream.
- You can plan coupling facility, log data set, and staging data set sizes to control how often offloading occurs.

The **high threshold** and offloading work differently for coupling facility versus DASD-only log streams:

- **For a coupling facility log stream**, system logger off-loads data from the coupling facility to DASD log data sets when coupling facility usage for the log stream reaches the high threshold. The high threshold is the point, in percent, of coupling facility structure usage for a log stream, where system logger will begin offload processing.
- **For a DASD-only log stream**, system logger off-loads data from local storage buffers to DASD log data sets. The high threshold, however, is the point, in percent, of staging data set usage. Thus, for a DASD-only log stream, offload processing moves log data from local storage buffers to DASD log data sets, but the offloading is actually triggered by staging data set usage.

Note: Other events exist that can trigger an offload. For more information, see [“Other events that trigger offloading”](#) on page 225.

For either type of log stream, when a log stream reaches or exceeds its high threshold, system logger begins processing to offload enough of the oldest log stream data to get to the low offload threshold point specified in the log stream definition. Note that a log stream might exceed the high threshold before

offloading starts, because applications might keep writing data before system logger can begin offload processing.

The **low threshold** is the target where offloading is to stop, leaving roughly the specified percentage of log data for that log stream in the coupling facility or staging data set.

For a **coupling facility log stream**, the high threshold you define for the coupling facility structure also applies to the staging data sets for the log stream; offloading is performed when either a staging data set or the coupling facility structure for a log stream hits its high threshold.

When an offload is triggered, each system connected to a log stream competes to gain ownership of processing the offload. This is controlled internally by system logger.

Note: Only one offload can occur at a time for a given log stream, regardless of the number of systems connected to it.

While offload processing is occurring, system logger applications can still write data to the log stream; system logger accepts and completes write requests while offloading is in progress. Since system logger calculates the amount of data to offload at the time offload processing starts, by the time offloading has finished, the amount of data in the coupling facility or staging data set exceeds the target low threshold.

The system that "wins" this competition then commences the offload process for the log stream. The offload process is bounded by the log stream's defined HIGHOFFLOAD and LOWOFFLOAD thresholds. Historically, offload processing comprises the following multiple phases:

Delete-only phase

Any data marked for deletion by the exploiter can be physically deleted from the interim storage medium without movement to DASD offload data sets. All log data that resides in interim storage that falls into this category is physically deleted, regardless of the log stream LOWOFFLOAD threshold. The only exception is when a RETPD value other than zero (0) is defined for the log stream. In that case, all the log data in the interim storage is moved to DASD before any physical deletion can occur.

Movement phase

Following the delete-only phase, if the amount of physical space consumed in interim storage is still above the LOWOFFLOAD threshold, system logger determines that it needs to begin moving data from the interim storage medium to the log stream secondary storage medium, for example DASD offload data sets.

This movement process brings in the additional complexity of managing these DASD offload data sets, and has the following multiple sub-phases:

- When the movement phase begins, the offloading system must first determine whether the current offload data set (that is, the data set currently available for logger to write into) is already allocated and available on the system. If not, a request is initiated internally to drive a logger task to allocate and open the offload data set.
- Once the current offload data set is allocated and ready for use, the offloading system will begin reading from the interim storage medium, for example CF, and writing the corresponding log data to the current offload data set. As logger confirms that the data was written successfully, it can physically delete the hardened data from interim storage. This process is followed until the LOWOFFLOAD threshold target is reached.
- Periodically, the current offload data set fills up. The frequency at which this occurs is based on various factors, including the size of the data set and the amount of data being moved. When this occurs, a current offload data set switch is initiated. This consists of:
 - Deallocating and closing the current offload data set.

Note: After this, it is still available for read processing, but can no longer be written into.

 - Creating a new offload data set. This is done by allocating a data set new,keep, then deallocating it. It is then allocated again with the SHR disposition and opened for processing.

Offload data set switch processing occurs as often as is needed to contain the valid log data being moved from interim storage. Due to the various data set allocations and requests utilized to prepare each new data set, it must run under a logger task instance that is dedicated to handling

offload related data set allocations. These factors also make data set switch processing essential for keeping log stream resources available, as various DASD considerations and constraints can come into play. If logger cannot obtain a new current offload data set for the log stream, the interim storage resource eventually completely fills, which leads to logger marking the log stream unavailable for new log data.

Offload data set delete phase

Offload data sets that are eligible for deletion might be physically deleted at this point. This phase can occur at various points during the offload based on certain factors, such as if Logger detects that it is constrained on directory entries.

For more information about when log data sets become eligible for physical deletion, see [“Deleting log data and log data sets” on page 291](#).

When the offload completes, system logger issues an ENF 48 signal.

The LS_ALLOCAHEAD log stream attribute can assist in minimizing the impact of offload data set allocations and related problems that can inhibit offload progress. This keyword enables logger to keep an available queue of up to three (advanced-current) offload data sets ready for offload processing.

When a nonzero value is specified for LS_ALLOCAHEAD, systems with an IXGCNFxx parmlib policy statement of ALLOCAHEAD(YES) behave as described later. See the IXGCNFxx parmlib member in *z/OS MVS Initialization and Tuning Reference* for more information about the MANAGE OFFLOAD ALLOCAHEAD specification. See [“LOGR keywords and parameters for the administrative data utility” on page 358](#) and *z/OS MVS Programming: Assembler Services Reference IAR-XCT* for additional details about the log stream LS_ALLOCAHEAD attribute.

When the LS_ALLOCAHEAD value is (inclusively) between 1 and 3, it means all systems that are connected to and performing offloads for the log stream should be proactive in newly allocating up to the intended number of advanced-current offload data sets. It also indicates that these systems are proactive in opening (to make as ready as possible) the current offload data set and the first advanced-current offload data set.

The following characteristics are true of offloading systems:

- Asynchronous to the offload, an offloading system attempts to make ready the current offload data set, for example by allocating SHR and opening it, in preparation for the next offload activity.
- In parallel with the offload, an offloading system attempts to acquire the number of advanced-current, for example newly allocated, offload data sets. In addition, the first advanced-current offload data set is allocated again with the SHR disposition and then opened for write processing when it is needed by the offload thread.
- Normal data set switch processing becomes a "fast path" request:
 - Logger will still deallocate the existing current offload data set as described previously.
 - A new advanced-current offload data set is already allocated and open, and therefore available for use. This means that a significantly cheaper data set switch occurs in-line during the offload.

Note: If the offload movement phase processing ever catches up to allocate ahead processing, meaning that there is no next advanced-current offload data set available for use, the previously asynchronous data set request becomes synchronous. The offload must wait for it before resuming movement of log data to DASD.

Other non-offloading systems that connect to the log stream and specify ALLOCAHEAD(YES) are notified when the offload completes. At that point, if needed, those systems allocate SHR and open the then current offload data set. This ensures that the connected systems have the data sets ready should they gain control of the next or subsequent offload for the log stream.

Systems that specify ALLOCAHEAD(NO) in the IXGCNFxx parmlib policy behave as follows:

An offloading system

- Still needs to allocate new current offload data sets on an as-needed basis.
- Data sets already created for the log stream as advanced-current offload data sets can be used.

- However, these systems will not pro-actively allocate/open any advanced- current offload data sets created for the log stream.

Other non-offloading systems

These systems do not proactively allocate or open ahead any new current offload data sets.

LS_ALLOCAHEAD is also intended to assist the installation by providing earlier awareness of problems that might impede an offload. By allocating data sets before they are needed, errors can be surfaced earlier, giving the installation time to address the problem.

It should be noted that even though the LS_ALLOCAHEAD option can be specified for group PRODUCTION and TEST log streams, the logger offload data set management is better optimized for PRODUCTION group log streams.

There is a set of messages that serve as indicators that Logger is having problems allocating offload resources. See [“Offload and service task monitoring”](#) on page 227, which identifies messages and specific actions which can be taken to resolve hung (or at least excessively delayed) log stream offloads, for more information.

Other events that trigger offloading

In general, normal offloading is triggered when the high offload threshold for coupling facility or staging data set usage for a log stream is reached. However, there are other system events that cause system logger to begin offload processing. In these cases, all the log stream is offloaded. Events triggering offload include the following:

- Recovery processing for a log stream. When recovery for a log stream is complete, system logger flushes all log data to DASD. See [“System logger recovery”](#) on page 300 for details.
- Structure rebuild, for coupling facility log streams.
- When the last connector to a log stream disconnects.
- Entries in-use on the structure reach 90%.

Another event that triggers offloads is through the LOCALBUFFERLIMIT option. This option triggers offloads when local buffers is almost exhausted. For more information about the MANAGE OFFLOAD LOCALBUFFERLIMIT specification, see [IXGCNFxx \(system logger initialization parameters\)](#) in the *z/OS MVS Initialization and Tuning Reference*.

Using the high and low thresholds to control offloading

Applications can influence the frequency of offloads by choosing high and low offload thresholds on the log stream definition in the LOGR, LOGRY or LOGRZ policy on the HIGHOFFLOAD and LOWOFFLOAD parameters.

In general, the greater the difference between the two thresholds the less often offloading occurs. If you choose threshold values that are close together, it means that you can keep more data on the coupling facility or local storage buffers for high-performance, low-overhead reads of the log data, but it also means that offloading and offloading overhead will occur more often.

The default thresholds are 80% for HIGHOFFLOAD and 0% for LOWOFFLOAD. This is a good setting for applications that primarily write to the log stream and do not retrieve data often, such as logrec log stream and OPERLOG. This setting has the advantage of moving larger chunks of log data to DASD at one time, reducing the number of times that offloading runs.

On the other hand, applications that want a relatively high amount of their log data to reside in the coupling facility or local storage buffers for quick access, transaction logs, for example, might find a HIGHOFFLOAD threshold of 80% and a low offload threshold of 60% more desirable.

IBM suggests that you do not define your HIGHOFFLOAD value to greater than the default of 80%. If you define a higher HIGHOFFLOAD value, you will be more vulnerable to filling your coupling facility or staging data set space for the log stream during sudden bursts of system activity that push the log data above the high threshold. Once coupling facility or staging data set space for a log stream is 100% filled, system logger rejects all write requests until the log data can be off-loaded to DASD log data sets.

In certain configurations, the log stream and related resources may be configured to try to avoid physically moving log data during the offload process - that is, the offload would only delete data from interim storage and not move it to DASD. This configuration is an approach to avoiding the most common offload inhibitors which involve the obtaining of DASD resources. In this configuration, the HIGHOFFLOAD and LOWOFFLOAD parameters play a key role in ensuring active data is retained in interim storage and offload threads only active on logically deleted log data. For more information on this configuration, see [“Monitoring log stream interim storage consumption” on page 228](#)

In any case, it is important to consult the documentation on the log manager application you are running for its application-specific recommendations.

You can tune your threshold settings by monitoring each log stream through SMF data, which shows:

- The number of times offloading has run.
- The number of coupling facility structure full conditions (for coupling facility log streams).
- The number of staging data set full conditions.
- The number of bytes written to and deleted from interim storage (coupling facility for coupling facility log streams or staging data set for DASD-only log streams).

For more information on SMF88 data, see: Record Type 88 (58) – System Logger Data in [z/OS MVS System Management Facilities \(SMF\)](#).

Offloading and log data deletion

Physical deletion of log data that is marked for deletion is tied to offloading; system logger does not physically delete the log data or log data set until an offload occurs. For more information, see [“When is my log data or log data set physically deleted?” on page 292](#).

Coupling facility size and offloading

For a coupling facility log stream, applications can influence the frequency of off-loads and the processing overhead offloading entails by sizing their coupling facility structure carefully. The larger the coupling facility structure, the less often offloading processing will occur and the larger the chunk of data that will be off-loaded. This minimizes offloading overhead and keeps more log data available on the coupling facility structure for high performance reads of log data.

For example, if you have a log stream with a high volume of data being written to it, and you have allocated a small coupling facility structure for this log stream, the coupling facility structure will fill quickly and meet its high threshold, initiating frequent offloading of log data to DASD. This increases system usage, but does not interfere with processing of write requests to the log stream, because write requests can run at the same time as offload processing.

However, the coupling facility structure for a log stream should not be larger than the application really requires and should reflect the needs of the rest of the installation for coupling facility space. See [“Determine the size of each coupling facility structure” on page 255](#) for information about optimal sizing of coupling facility structures for response, throughput, and availability.

If the coupling facility space allocated for a log stream reaches 100% utilization, all write requests against that log stream are rejected until offloading can complete.

Staging data set size and offloading

Coupling facility log stream

For a coupling facility log stream, the high offload threshold defined by an installation for the coupling facility space allocated to a log stream applies also to the staging data sets in use by connections to that log stream. For example, if you define a high threshold of 80% for a log stream, system logger will begin off-load processing when either the coupling facility structure allocated for a log stream or the staging data set for any connection to the log stream reaches or exceeds 80% usage.

This means that when a staging data set reaches the high threshold, system logger immediately off-loads data from the coupling facility to log data sets, even if the coupling facility usage for the log stream is below the high threshold.

Thus, if your staging data sets are small in comparison to the coupling facility structure size for a log stream, the staging data sets will keep filling up and system logger will offload coupling facility data to DASD frequently. This means that your installation experiences frequent offloading overhead that could affect performance.

If your staging data sets are too small, you also run the risk of filling them up completely. If this occurs, system logger immediately begins offloading the coupling facility log data in DASD log data sets to harden it. System logger applications will be unable to log data until system logger can free up staging data set space. See the recovery topic in the system logger chapter of *z/OS MVS Programming: Assembler Services Guide*.

In general, the sizing guideline for staging data sets is to make them large enough to hold all the log data in the coupling facility. See [“Plan space for staging data sets”](#) on page 263 for complete sizing staging data sets.

DASD-only log streams

For a DASD-only log stream, offloading of log data to DASD log data sets is always triggered by staging data set usage. System logger off-loads data from local storage buffers to DASD log data sets when it hits the high threshold defined for the staging data sets. For example, if you define a high threshold of 80% for a DASD-only log stream, system logger will begin offload processing when the staging data set space defined for a log stream reaches or exceeds 80% usage.

This means that if you size your staging data sets too small, system logger will offload log data from local storage buffers to DASD log data sets frequently, incurring overhead. You might also run the risk of filling the staging data sets up completely. If this occurs, system logger will immediately begin offloading log data to DASD log data sets.

Offload and service task monitoring

System logger monitors (in 5 second intervals) log-stream-offload activity as well as specific log stream data set allocation requests and notifies the installation if an offload appears to be hung or is taking too long by issuing messages, such as IXG312E, IXG272E, IXG281I, or IXG115A. If this happens, there may be several system logger and non-system logger messages for which responses are expected.

Consider the following steps when responding to these messages:

1. Determine if there are inhibitors to offload processing by issuing the following commands; then correct any problems:

```
D  LOGGER,C,LSN=logstreamname,DETAIL
D  LOGGER,L,LSN=logstreamname
D  LOGGER,STATUS,RECALLS
D  XCF,STRUCTURE,STRNAME=structname
D  GRS,C
```

2. If the problem persists, respond to any allocation, catalog, or recall messages such as IEF861I, IEF863I, or IEF458D.

If the DISPLAY LOGGER,STATUS,RECALLS results in message IXG601I revealing that log stream data set recalls are not being satisfied, consider using the SETLOGR FORCE,NORECALL command to cause system logger to stop waiting for a particular data set to be recalled.

If system logger message IXG272E is displayed, reply to this message only after you have attempted to correct any general allocation- or recall- related messages.

If message IXG271I indicates that the delay affects a log stream data set and you reply "FAIL" to the accompanying message IXG272E, log stream duplexing might be affected and the log stream connection might not be able to continue. A reply should only be made when all other avenues have been exhausted.

3. If the problem persists, respond to any IXG312E messages. This can be used to stop the offload processing for the log stream named in the message and allow it to run on another system, if possible. It might allow other work to run on the system that was attempting the original offload.
4. If message IXG115A is displayed, reply to this message only after you have attempted to correct any delayed off-loads by responding to the related IXG312E and IXG272E messages. As a last resort, if you reply "TASK=END" to an IXG115A message, system logger ends all the log stream connections in the structure named in the message on this system.

Review the complete description of messages IXG310I, IXG311I, IXG312E, IXG271I, IXG272E, IXG281I, IXG114I, and IXG115A in [z/OS MVS System Messages, Vol 10 \(IXC-IZP\)](#) before responding to any of these messages.

Note that several messages could appear at the same time for different log streams for which offloading is taking place. It may be that only one log stream is actually having a problem, and the others are simply waiting for this log stream to finish its allocation processing. Usually, the log stream that is causing the delay is the first log stream to be reported on.

Another way to determine which log stream is actually causing the delay is to check to see if the data set name ends with an explicit sequence number, such as "A0000001". If this is the case, it usually means that this log stream is experiencing a problem. If the data set name ends with "<SEQ#>", then it is more likely that this log stream is waiting for another log stream that is having a problem.

Additionally, as interim storage fills, additional messages may be triggered based on the system logger configuration. These messages are intended to monitor interim storage consumption and alert the installation when the capacity reaches critical points. Since offload processing is the method by which interim storage is made available for new data, these are closely related topics. See ["Monitoring log stream interim storage consumption"](#) on page 228 for further details.

System logger log stream offload activity is monitored using two sets of 2 intervals. The first set of intervals pertain to attempts to allocate or delete log stream offload data sets. The second interval set aids in monitoring migrated data set recall requests during the offload. Within each offload monitor set there is an initial (warning) interval and a secondary (action interval). Logger uses the following default offload monitoring intervals:

- For data set allocation/deletion requests, an initial interval of 30 seconds and secondary interval of 60 seconds (1/2 minute and 1 minute, respectively), and
- For a recall of a migrated data set request, an initial interval of 60 seconds and secondary interval of 120 seconds (1 minute and 2 minutes, respectively).

When an initial log stream offload monitor interval is reached, system logger will issue message IXG310I to provide an early warning of a delay. When the secondary log stream offload monitor interval is reached, logger issues messages IXG311I and IXG312E to allow possible action on the delayed offload. These two interval values can be specified by the installation. See [z/OS MVS Initialization and Tuning Reference](#) for information on providing installation specifications for these log stream offload monitoring intervals in SYS1.PARMLIB member IXGCNFxx.

Monitoring log stream interim storage consumption

System logger internally tracks the usage of the log stream interim storage medium. Starting at z/OS V2R1, key information can be revealed to both the installation and exploiter, depending on the configuration of IXGCNF and log stream parameters.

This information can assist in exposing offload inhibitors, including those inhibitors which may not trigger logger offload or task monitoring alerts – such as an abnormal DASD I/O delays.

There are different sets of system logger parameters that affect the level of monitoring for a log stream on a given z/OS system.

The first set of parameters to configure are in the log stream definition in the LOGR, LOGRY or LOGRZ policy. They are the WARNPRIMARY and HIGHOFFLOAD parameters. System logger establishes an imminent threshold for each log stream based on an interim storage usage percentage above HIGHOFFLOAD. The imminent threshold value is 2/3rd of the difference between 100% full and the

HIGHOFFLOAD value. The WARNPRIMARY parameter indicates whether warning messages for this log stream should be issued when the imminent threshold is reached or a 90% entry full condition is encountered for structure based log streams. The default WARNPRIMARY value is NO.

These parameters have the following effects:

- When the imminent threshold is reached, regardless of the WARNPRIMARY parameter, the IXGWRITE ANSAA field ANSAA_WRITEIMMINENTCAPACITY will be set on. See *z/OS MVS Programming: Assembler Services Guide*, for more details.
- When the imminent threshold is reached and WARNPRIMARY(YES) has been specified, message IXG317E will be issued to the console to alert the installation that the primary storage consumption has reached the threshold.
- When a 90% structure entry full condition is encountered and WARNPRIMARY(YES) has been specified, message IXG316E will be issued to the console to alert the installation that the primary storage consumption has reached the entry near full threshold.

Note: This condition only occurs for log streams defined to use a CF structure.

- When an interim storage full condition is encountered and WARNPRIMARY(YES) has been specified or when a 100% structure entry full condition is encountered, message IXG318E will be issued indicating the interim storage resource is full.

The system logger alert messages remain in the retrievable messages list until the appropriate circumstances occur for each:

- The IXG318E constraint condition will be deleted when one of the following occurs:
 - After a log stream offload activity, system logger detects the log stream primary (interim) storage consumption appears to be at least 5% less than full, then message IXG319I will be issued and the system will delete message IXG318E.
 - When there are no longer any log stream connections on the system.
- The IXG317E imminent threshold message will be deleted when one of the following occurs:
 - After a log stream offload activity, system logger detects the percent in use falls at least 5% below the imminent threshold established for the log stream.
 - When the condition is superseded by a primary (interim) storage full condition and message IXG318E is issued for the log stream.
 - When there are no longer any log stream connections on the system.
- The IXG316E entry near full threshold message will be deleted when one of the following occurs:
 - After a log stream offload activity, system logger detects the percent in use falls at least 85% or lower for the entire CF structure.
 - When the condition is superseded by a primary (interim) storage full condition and message IXG318E is issued for any log stream in the CF structure.
 - When there are no longer any log stream connections on the system using the CF structure.

The second set of parameters allows these messages to be enabled or disabled on a z/OS system basis. This is done through the IXGCNF CONSUMPTIONALERT parameter. The default setting for this parameter is ALLOW, and the interim storage usage alert messages will be issued at the previously described points. When the parameter is set to SUPPRESS, it indicates that no interim storage warning messages will be issued, regardless of the log stream parameters.

Additionally, XES provides structure utilization monitoring which can be valuable for log streams defined to use CF structures. The FULLTHRESHOLD parameter for a CF structure specification in the CFRM policy can be used to provide consumption warnings via z/OS XES messages (that is IXC585E and IXC586I). Message IXC360I also contains significant CF structure use and consumption information for a DISPLAY XCF,STR,STRNAME=*structurename* command. For more information, see [“Monitoring structure utilization” on page 60](#).

Note: These settings are critical in cases where the log stream and related resources may be configured to try to avoid physically moving log data during the offload process - that is, the offload would only delete

data from interim storage and not move it to DASD. This configuration is an approach to avoiding the most common offload inhibitors which involve the obtaining of DASD resources. For more information on this configuration, see [“Managing log data in interim storage to minimize movement to DASD”](#) on page 253

When interim storage consumption reaches a critical point, look for indications that offload processing is not running in a normal manner:

- Review the complete description of any threshold message issued, such as IXG316E, IXG317E or IXG318E.
- Look for offload and service task monitoring related messages, such as IXG271I, IXG272E, IXG311I or IXG312E. See [“Offload and service task monitoring”](#) on page 227.

For more information, see:

- [“Offload and service task monitoring”](#) on page 227
- [“DEFINE LOGSTREAM keywords and parameters”](#) on page 359
- [“Offloading log data from interim storage by freeing and/or moving it to DASD”](#) on page 222
- [z/OS MVS Initialization and Tuning Reference](#)

System clock considerations

System logger uses the system clock GMT value as an authorization key when writing to the coupling facility on behalf of the log stream.

If you change the GMT, specifically, turn the clock back, system logger will not be able to write to the log stream until the new GMT is greater than the old GMT. When changing the GMT, you should delete all of the log streams and then redefine them, especially if the GMT is moved backwards.

Changing the GMT affects both coupling facility and DASD-only log streams.

Example 1: A coordinated timing network (CTN) is a collection of servers and Coupling Facilities that are time synchronized to a time value called coordinated server time. A mixed-coordinated timing network (CTN) requires a Sysplex Timer and allows for both an external time reference (ETR) network and a server time protocol (STP).

There are circumstances when the TOD clock might drift enough for system logger to detect that a system on an ETR-timing CPC may be less advanced than one on the STP-timing CPC. The drift can occur based on how long the stratum 2 server resides in STP-timing mode during reverse migration from Mixed-CTN to an ETR-only network on z/OS images on the CPC for which the ETR ports are enabled.

The TOD clock drift might result in system logger issuing an ABEND01C5, rsn00040003 and a dump is taken to record the condition. System logger continues operating without any other problems, but log data cannot be written into the log stream from this system until the condition is cleared. See [z/OS MVS System Codes](#) for System Action and System Programmer Response for this condition.

Example 2: Log stream “IXGLOGR.TEST” is an active log stream and contains data written by an application using system logger services. The system is IPLed. The last block written, before the IPL, was written at 13:00:00 Mar 1, 2000. This date and time is the authorization key used by system logger. During the IPL, the GMT is changed to 13:00:00 Mar 1, 1998. When system logger attempts to write the next block to IXGLOGR.TEST, it will use the new GMT. System logger will detect the difference in the GMT values and abend.

Note: For a fail-safe means of preventing such problems, delete the log stream while running on the test system, and then redefine the log stream on the system. If you IPL the production system without having deleted the log stream previously, you might run into production problems trying to delete the log stream on the production system. Message IXG212E (RECOVERY FOR LOGSTREAM *logstream* IN STRUCTURE *strname* WAS NOT SUCCESSFUL) has been seen as a symptom when attempting to IPL a production system again without having done the delete on the test system.

Logger configuration resource value ranges/limits

Table 14 on page 231 contains a list of the system logger resources with a description and a summary of the limits or value ranges for each entity. Parallel sysplex or base sysplex considerations are noted where applicable.

Table 14. Summary of logger resources			
Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
LOGR CDS (couple data set, sysplex scope)	LOGR	<p>Contains system logger policy and log stream resource inventory and state information on a sysplex basis.</p> <p>See “Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276 for considerations on establishing the active set of LOGR couple data sets as well as having spares available.</p> <p>Also see XCF program utilities IXCL1DSU and IXCMIAPU, and the SETXCF COUPLE command.</p>	<p>Only one active primary LOGR CDS in a sysplex.</p> <p>One alternate LOGR CDS also allowed to be active (which is a copy of primary in ready state in case of SETXCF command requesting a PSWITCH).</p>
LOGRY CDS (couple data set, single-system scope)	LOGRY	<p>Contains system logger policy and log stream resource inventory and state information on a single-system basis.</p> <p>See “Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex” on page 279 for considerations on establishing the active set of LOGRY couple data sets as well as having spares available.</p> <p>Also see XCF program utilities IXCL1DSU and IXCMIAPU, and the SETXCF COUPLE command.</p>	<p>Only one active primary LOGRY CDS in a sysplex and can only be in use on one z/OS image.</p> <p>One alternate LOGRY CDS also allowed to be active (which is a copy of primary in ready state in case of SETXCF command requesting a PSWITCH).</p>

Table 14. Summary of logger resources (continued)

Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
LOGRZ CDS (couple data set, single-system scope)	LOGRZ	<p>Contains system logger policy and log stream resource inventory and state information on a single-system basis.</p> <p>See “Using LOGRZ or LOGRZ couple data sets for a single system scope within a sysplex” on page 279 for considerations on establishing the active set of LOGRZ couple data sets as well as having spares available.</p> <p>Also see XCF program utilities IXCL1DSU and IXCMIAPU, and the SETXCF COUPLE command.</p>	<p>Only one active primary LOGRZ CDS in a sysplex and can only be in use on one z/OS image.</p> <p>One alternate LOGRZ CDS also allowed to be active (which is a copy of primary in ready state in case of SETXCF command requesting a PSWITCH).</p>
LSR records	LOGR LOGRY LOGRZ	<p>System logger records in active CDS that determine overall number of log streams allowed to be defined for the sysplex.</p> <p>See “LOGR parameters for format utility (sysplex scope)” on page 320 and “LOGRY or LOGRZ single-system scope parameters for format utility” on page 326 for more details.</p>	32,767
LSTRR records	LOGR	<p>System logger records in sysplex scope LOGR CDS that determine overall number of CF structures allowed to be defined for logger use.</p> <p>See “LOGR parameters for format utility (sysplex scope)” on page 320 for more details.</p>	32,767

Table 14. Summary of logger resources (continued)

Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
DSEXTENT records	LOGR LOGRY LOGRZ	<p>System logger records in logger CDS that determine overall number of log stream data set directory extent records (allows for 168 offload data sets per DSEXTENT).</p> <p>See “LOGR parameters for format utility (sysplex scope)” on page 320 and “LOGRY or LOGRZ single-system scope parameters for format utility” on page 326 for more details.</p> <p>IBM recommends that you plan for DSEXTENT records to be defined in your active primary and alternate system logger couple data sets.</p>	99,998
SMDUPLEX record	LOGR	<p>System logger designation of format level for LOGR CDS. Also indicates whether system logger supports Coupling Facility system-managed duplexing.</p> <p>See “LOGR parameters for format utility (sysplex scope)” on page 320 for more details.</p>	<p>The HBB7705 format level is used when the LOGR CDS is formatted with the NUMBER(1) specification or when the SMDUPLEX option is omitted. IBM recommends using this setting to get the highest functional capabilities in system logger.</p> <p>The HBB6603 format level is used when the LOGR CDS is formatted with the NUMBER(0) specification.</p>
FMTLEVEL record	LOGRY LOGRZ	<p>System logger designation of format level for LOGRY and LOGRZ CDS data types</p> <p>See “LOGRY or LOGRZ single-system scope parameters for format utility” on page 326 for more details.</p>	<p>The HBB77C0 format level is used when either the LOGRY CDS or LOGRZ CDS is formatted with the NUMBER(1) specification or when the FMTLEVEL option is omitted.</p> <p>IBM recommends using this setting to get the highest functional capabilities in system logger.</p>
Number of connectors to a single log stream -- any type	LOGR LOGRY LOGRZ	Concurrent connection instances from exploiters that connect to & share (either read or write) data in a single log stream.	No specific limit, but system memory usage will govern allowance per system.

Table 14. Summary of logger resources (continued)

Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
Number of systems connected to a log stream	LOGR LOGRY LOGRZ	z/OS images that can concurrently connect to a single log stream (based on type)	Up to 32 systems concurrently for CF structure based log stream (given # systems in the Parallel sysplex, re: MAXSYSTEM value for DEFINEDS statement on IXCL1DSU utility). Only 1 system can connect to DasdOnly based log stream at a time.
Number of log streams in a single CF structure	LOGR	Log streams mapped (meaning defined) to a CF structure resource.	512 But the recommendation is to limit to 10 or less since it can have significant impact of CF structure resources use and log data recovery operations.
Number connected CF structures for logger GROUP(PRODUCTION)	LOGR	Number of logger CF structures with production group log stream(s) that can be connected concurrently.	255 on one z/OS image. 1024 for sysplex (assuming the CF allocated limit).
Number connected log streams in CF structures GROUP(PRODUCTION)	LOGR	CF structure based production group log streams that can be connected concurrently.	32,767 on one z/OS image (given LSR number limit). But system memory usage will govern actual allowance per system. 32,767 for sysplex (given the LSR number limit).
Number connected CF structures for logger GROUP(TEST)	LOGR	Number of logger CF structures with test group log stream(s) that can be connected concurrently.	63 on one z/OS image. 1024 for Parallel sysplex (assuming CF allocated limit).
Number connected log streams in CF structures GROUP(TEST)	LOGR	CF structure based test group log streams that can be connected concurrently.	32,256 on one z/OS image (given number of test CF structures and number of log streams in a single CF structure). But system memory usage will govern actual allowance per system. 32,767 for Parallel sysplex (given the LSR number limit).

Table 14. Summary of logger resources (continued)

Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
Number connected DasdOnly log streams GROUP(PRODUCTION)	LOGR LOGRY LOGRZ	DasdOnly based production group log streams that can be connected concurrently.	16,384 on one z/OS image. But system memory usage will govern actual allowance per system. 32,767 for sysplex (given the LSR number limit).
Number connected DasdOnly log streams GROUP(TEST)	LOGR LOGRY LOGRZ	DasdOnly based test group log streams that can be connected concurrently.	4096 on one z/OS image. But system memory usage will govern actual allowance per system. 32,767 for sysplex (given the LSR number limit).
Size of a "log block" object in a log stream	LOGR LOGRY LOGRZ	Maximum log block size in a log stream, which is the data unit that can be atomically read/written by logger operations.	64K-4 bytes (65,532). But can be limited by the MAXBUFSIZ keyword on a STRUCTURE or LOGSTREAM definition.
Staging data set size	LOGR LOGRY LOGRZ	Maximum size of an individual log stream staging data set (VSAM linear data set).	Up to 16TB. For example, 16384GB or 17,592,186,044,416 bytes. For system levels prior to V2R3, up to 4 GB. IBM recommends that you plan for staging data sets for each system even if your installation does not intend to use them for duplexing and does not define duplexing to staging data sets in the log stream definition. IBM recommends that staging data sets be sized no smaller than 10 MB, unless the exploiter specifically recommends a size below this amount.
Offload data set size	LOGR LOGRY LOGRZ	Maximum size of an individual log stream offload data set (VSAM linear data set).	Up to 4 GB. IBM recommends that you size offload data sets no smaller than 1 MB, unless the exploiter specifically recommends a size below this amount.

Table 14. Summary of logger resources (continued)

Logger Resource / Entity	Relevant CDS data types	Description	Configuration Limit / Value range
Number of staging data sets	LOGR LOGRY LOGRZ	Number of DASD staging data sets for a single log stream, when duplex configuration deems it necessary.	One (1) staging data set per system (i.e. z/OS image) connected to a log stream.
Number of offload data sets	LOGR LOGRY LOGRZ	Number of DASD offload data sets for a single log stream.	168 data set instances, plus 168 more per DSEXTENT record that is assigned (added) to the log stream's overall data set directory.

Considerations for encrypting log stream data sets

Data set encryption can be used for log stream data sets to enhance an installation's capability in securing their sensitive enterprise data. Both log stream offload and staging data set types can have data set level encryption.

The system programmer installs and applies the appropriate release levels and any required PTFs, and ensures that the necessary hardware features are established. Then, the security administrator works with the ICSF administrator and storage administrator to set up policies that enable the log stream data set encryption. For more information about using data set level encryption, refer to "Data Set Encryption" in *z/OS DFSMS Using Data Sets*.

Log stream data set encryption can be enabled through DFSMS data class definitions (or ACS routines), or through RACF data set profiles by naming KEYLABELS that define cryptographic keys to be used for data set level encryption.

IBM recommends that all systems in the sysplex must be at the appropriate release and/or PTF levels before enabling log data set encryption for log stream resources.

Limitations and special considerations:

Some limitations and special considerations should be examined before enabling data set level encryption for log stream data sets.

1. Multiple log streams can be defined to use the same CF structure, and separate z/OS images can connect to some but not all of these log streams. If some of these z/OS images are not at the appropriate release and/or PTF levels for encryption support, then do not enable data set level encryption for any of these log streams. System logger, on any system that is connected to a CF structure, recovers log stream data that is contained in that CF structure. z/OS systems without the appropriate support applied, encounter data set allocate and open access errors if/when they attempt to recover for a log stream that has encrypted data sets.
2. System Logger does not support encrypted data sets for any DRXRC-type staging data sets. A structure-based log stream that has STG_DUPLEX DUPLEXMOD(DRXRC) specified as an allocated encrypted staging data set results in an open error when system logger attempts to use it.
3. When a z/OS V2R1 release level system (with PTF applied) is connected to a log stream with encrypted data sets, system logger is able to access the log data that is contained in the encrypted data sets. However, these z/OS V2R1 systems cannot create any new encrypted log stream data sets. The installation needs to make certain any new log stream data sets created on the V2R1 system are not expected to be encrypted, for example: using an alternative DFSMS data class without a data set key label. Some of the log stream data sets can be encrypted, but any data sets created on the V2R1 systems would not be encrypted. Otherwise, the log stream can be used in a limited fashion only, with key aspects listed below:

- Initial log stream connections on the V2R1 system that would normally cause the creation of a log stream staging data set can fail. This occurs for DASDONLY log streams and CF structure-based log streams with STG_DUPLEX(YES) DUPLEXMODE(UNCOND).
- Log stream offload activity on the V2R1 system that requires the creation of an extra offload data set, will not succeed. This occurs during log stream log data recovery processing and/or during normal log stream use. The log stream offload activity on the V2R1 system performs a log stream offload if no new offload data sets were to be created. Also, other systems at the appropriate levels would continue to manage the log stream as expected.

Minimum requirements for encrypting log stream data sets:

The following lists the minimum required actions necessary to enable encrypting log stream data sets.

- The Integrated Cryptographic Service Facility (ICSF) must be configured as follows. ICSF requires Cryptographic hardware to be available on the server. For details, refer to [z/OS Cryptographic Services ICSF Administrator's Guide](#) and [z/OS Cryptographic Services ICSF System Programmer's Guide](#).
 - Activate the ICSF address space (mandatory).
 - Set up and maintain cryptographic keys and the KEYLABELS to be used for data set level encryption.
 - Ensure that the Sysplex ICSF (database) repository is shared and maintained across all systems in the sysplex, and on any recovery site systems. For more information, refer to "Running in a Sysplex Environment" in [z/OS Cryptographic Services ICSF Administrator's Guide](#) and sysplex-related specifications of "Installation, initialization, and customization" in [z/OS Cryptographic Services ICSF System Programmer's Guide](#).
 - Ensure that the catalogs are also shared across these same systems when data set level encryption is used.
 - Update the ICSF segment of the covering CSFKEYS general resource class profile with the following new options to ensure that wrapped keys can be returned (mandatory):

SYMCPACFWRAP(YES)

Allows key label to be wrapped.

SYMCPACFRET (YES)

Allows the service to return the wrapped key.

- For protected keys
 - Set up profiles in the CSFKEYS general resource class and set up groups and users to allow system logger address space (IXGLOGR) access to the key labels. Refer to "Setting up profiles in the CSFKEYS general resource class" in [z/OS Cryptographic Services ICSF Administrator's Guide](#).
 - Also, refer to "Enabling use of encrypted keys in Symmetric Key Encipher and Symmetric Key Decipher callable services" in [z/OS Cryptographic Services ICSF Administrator's Guide](#).
 - The following RACF commands provide an example of the two previous requirements:

```
RDEFINE CSFKEYS * UACC(READ) ICSF(SYMCPACFWRAP(YES) SYMCPACFRET(YES))
SETR RACLIST( CSFKEYS) REFRESH
```

- Set up profiles in the CSFSERV general resource class and set up groups and users to allow system logger address space (IXGLOGR) access to the CSFKRR2 resource.
- Data set encryption can be established for SMS-managed data sets defined with the Extended format option, allocated on (at least) 3390 device types by:
 - Activating the SMS address space.
 - Ensuring to specify ACSDEFAULTS(YES) in SYS1.PARMLIB(IGDSMSxx) member.
- You can optionally use a key label in RACF DS profiles. For information about RACF, see [z/OS Security Server RACF Security Administrator's Guide](#).
 - Create RACF data set profiles. In the DFP Segment, specify the key label wanted (see DATAKEY) to cover log stream data sets.

- Or you can define DFSMS Data Classes with Extended format option by using the Interactive Storage Management Facility (ISMF) panels or via ACS routines. For information about ISMF, see [z/OS DFSMS Using the Interactive Storage Management Facility](#).
 - Optionally, include a specification for a data set key label in the data class definition.
 - To allow the system to create encrypted data sets via DFSMS data class or ACS routine, the user must have at least READ authority to the following resource in the FACILITY class:
STGADMIN.SMS.ALLOW.DATASET.ENCRYPT.
- Specify on the log stream definition, or update an existing definition with, the corresponding DFSMS data class names established previously for the log stream offload and staging data sets. Refer to LS_DATACLAS and STG_DATACLAS log stream keyword specifications.
- Take action for the subsystem or application to connect and write log data to the log stream. Any newly created log stream data sets that use the previous specifications will have data set level encryption.
- For similar steps and guidance on changing a log stream staging data set's attribute when the log stream is already connected, see [“Change the staging data set size for coupling facility log streams”](#) on page 264 and [“Changing the staging data set size for a DASD-only log stream”](#) on page 266.

Operational considerations:

When system logger requests an encrypted log stream data set to be opened, interaction with ICSF needs to occur to honor the open request. If ICSF is not yet available, for example after an IPL and it is not initialized yet or was brought down temporarily for service and was not yet restarted, then the open request would not be completed.

For cases when ICSF is unavailable during an attempt to open an encrypted log stream data set, system logger will notify operations with a WTOR message (IXG079E). System logger will wait for ICSF to become available (to be active) or for a reply to the message.

After a response to IXG079E is provided, system logger will issue message IXG080I indicating that it is no longer waiting on ICSF. While logger waits for this condition to be resolved, progress on other log stream requests can be impeded, and can potentially negatively impact other exploiters accesses to their log streams.

Update your IPL procedures to include getting ICSF started and for awareness on the proper handling of the IXG079E WTOR message.

Approaches to stop using log stream data set encryption:

Once log stream data set encryption is enabled and in use, you should not remove the logger functional capability to support data set encryption (meaning remove the support provided in this PTF) until you have taken the appropriate steps to enable acceptable access to existing log stream log data. If the logger capability to support data set encryption is removed from your system, then logger will not be able to manage/access and encrypt log stream data sets, and this will lead to failures for the log stream exploiters since they will not be able to access their log data.

To stop using log stream data set encryption, first stop newly created log stream data sets from being encrypted data sets. Then, copy any log data from the log stream to an alternative medium, such as an archive data set.

Next, you must delete existing log stream data sets. Depending upon each log stream exploiter, your options to cause the deletion of existing log stream data sets can vary greatly. For example, you might need to disconnect the log stream from all the connected systems in the sysplex, or you might need to delete the log stream entirely.

Based on how you enabled the encryption, you can update your log stream definition to use a different DFSMS data class, change the DFSMS data class definition or ACS routines, or update the RACF profiles that cover the log stream data sets. This stops system logger from creating new encrypted log stream data sets, but does not alter any existing encrypted data sets.

Finding information for system logger applications

System logger applications can be supplied by IBM, independent software vendors, or written at your installation. For each of these applications, some information is necessary for set-up and use.

Some of the IBM-supplied system logger applications are:

- **CICS log manager** is a CICS system logger application that replaces the journal control management function. It provides a focal point for all CICS system log, forward recovery log, and user journal output within a sysplex and flexible mapping of CICS journals onto log streams. CICS log manager also enables faster CICS restart, dual logging, and flexible log and journal archiving.
- **Logrec log stream** is an MVS system logger application that records hardware failures, selected software errors, and selected system conditions across the sysplex. Using a logrec log stream rather than a logrec data set for each system can streamline logrec error recording.
- **Operations log (OPERLOG)** is an MVS system logger application that records and merges messages about programs and system functions (the hardcopy message set) from each system in a sysplex that activates OPERLOG. Use OPERLOG rather than the system log (SYSLOG) as your hardcopy medium when you need a permanent log about operating conditions and maintenance for all systems in a sysplex.
- **APPC/MVS** uses a System Logger log stream whenever a synchronization level of SYNCPT is selected by a transaction program, and when an LU has been made syncpt-capable. This log stream is used to store persistent data needed in support of the two-phase commit protocol.
- **IMS Common Queue Server (CQS) log manager** is an IMS system logger application that records the information necessary for CQS to recover structures and restart. CQS writes log records into a separate log stream for each coupling facility list structure pair that it uses. IMS CQS log manager uses coupling facility based log streams.
- **RRS (resource recovery services)** is an MVS system logger application that records events related to protected resources. RRS records these events in five log streams that are shared by systems in a sysplex. Use the information in *z/OS MVS Programming: Resource Recovery* about defining the log streams with the information in [“Preparing to use system logger applications” on page 243](#) to set up and use the RRS system logger application.

Use the following topics to find the information you'll need to set up and use these system logger applications:

- [“Finding information for CICS log manager” on page 239](#)
- [“Finding information for OPERLOG log stream” on page 240](#)
- [“Finding information for logrec log stream” on page 241](#)
- [“Finding information for APPC/MVS” on page 242](#)
- [“Finding information for IMS common queue server log manager” on page 242.](#)
- [“Finding information for resource recovery services” on page 243](#)
- [“Finding information for system management facilities \(SMF\)” on page 243](#)

For writing your own system logger application using the system logger services, see

- [“Preparing to use system logger applications” on page 243](#)
- [z/OS MVS Programming: Assembler Services Guide](#)
- [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#)

To prepare to use a system logger application supplied by an independent software vendor, see [“Preparing to use system logger applications” on page 243](#).

Finding information for CICS log manager

Use [Table 15 on page 240](#) to find information to help you set up and use the CICS log manager. Almost all of the information you will need is in the CICS library, but there are a few steps needed to define CICS log streams to MVS itself. Note that it is important that you follow the CICS recommendations when using the CICS log manager.

Table 15. Finding CICS log manager information	
What is your goal?	You need to read:
Prepare for using the CICS log manager: <ul style="list-style-type: none"> • Learn about CICS log manager • Plan for and set up the CICS log manager log streams • Review log stream names and develop a naming convention for system logger resources • Size the coupling facility structure space needed for the CICS log manager. 	<ul style="list-style-type: none"> • <i>CICS Transaction Server for z/OS Release Guide</i> • <i>CICS Transaction Server for z/OS Installation Guide</i> • <i>CICS System Definition Guide</i> These titles and other CICS documentation are available at CICS Transaction Server for z/OS (www.ibm.com/docs/en/cics-ts) .
Plan and set up DASD space needed for CICS log manager log data sets and staging data sets.	<ul style="list-style-type: none"> • “Plan DASD space for system logger” on page 260 • <i>CICS Transaction Server for z/OS Installation Guide</i>
Define authorization to system logger resources for CICS log manager.	“Define authorization to system logger resources” on page 273
Define the CICS log manager DASD log and staging data sets to global resource serialization in the GRSRNLxx parmlib member.	“Add the DASD data sets to the GRSRNL inclusion list” on page 267
Format the LOGR couple data set, identify it to the sysplex, and make it available.	“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276
Add, update, or delete CICS policy data (CICS log streams and coupling facility structures) in the LOGR policy using the IXCMIAPU utility.	<ul style="list-style-type: none"> • “LOGR keywords and parameters for the administrative data utility” on page 358 • See <i>CICS Transaction Server for z/OS Installation Guide</i> for recommendations for structure and log stream attributes.
Update the CFRM sysplex couple data set with CICS log manager coupling facility structures.	<ul style="list-style-type: none"> • “CFRM parameters for administrative data utility” on page 346 • See <i>CICS Transaction Server for z/OS Installation Guide</i> for recommended structure and log stream attributes and sample jobs. • See <i>CICS Transaction Server for z/OS Release Guide</i> for sample jobs.
Activate the LOGR subsystem.	“Activate the LOGR subsystem” on page 291.

Finding information for OPERLOG log stream

Use [Table 16 on page 240](#) to find information to help you set up and use the OPERLOG system logger application.

Table 16. Finding OPERLOG information	
What is your goal?	You need to read:
<ul style="list-style-type: none"> • Learn about the operations log stream (OPERLOG). • Decide whether to use OPERLOG or SYSLOG as your hardcopy medium. • Plan to use OPERLOG as your hardcopy medium. 	z/OS MVS Planning: Operations
Learn requirements for system logger applications.	“Understand the requirements for system logger” on page 244.
Develop a naming convention for system logger resources.	“Develop a naming convention for system logger resources” on page 258

Table 16. Finding OPERLOG information (continued)	
What is your goal?	You need to read:
Do set-up for the OPERLOG log stream.	See “Preparing to use system logger applications” on page 243.
Size the coupling facility structure space needed for the OPERLOG log stream.	See the appendix on coupling facility structures in <i>System/390 MVS Parallel Sysplex Configuration Cookbook</i> , SG24-4706.
Tune the OPERLOG log stream definition using SMF record type 88.	z/OS MVS System Management Facilities (SMF)
Delete OPERLOG log data.	<ul style="list-style-type: none"> • “Managing log data: How much? For how long?” on page 272 • “Deleting log data and log data sets” on page 291
At IBM's request, gather system logger component trace data.	z/OS MVS Diagnosis: Tools and Service Aids

Finding information for logrec log stream

Use [Table 17 on page 241](#) to find information to help you set up and use the logrec log stream.

Table 17. Finding Logrec Log Stream information	
What is your goal?	You need to read:
<ul style="list-style-type: none"> • Learn about logrec error records. • Decide whether to use the logrec data set or logrec log stream as your logrec recording medium. • Plan to use the logrec log stream. 	z/OS MVS Diagnosis: Tools and Service Aids
Learn requirements for system logger applications.	“Understand the requirements for system logger” on page 244.
Develop a naming convention for system logger resources.	“Develop a naming convention for system logger resources” on page 258
Do set-up for the logrec log stream.	See “Preparing to use system logger applications” on page 243.
Size the coupling facility structure space needed for the logrec log stream.	See the appendix on coupling facility structures in <i>System/390 MVS Parallel Sysplex Configuration Cookbook</i> , SG24-4706.
Activate the LOGR subsystem.	“Activate the LOGR subsystem” on page 291.
Use the LOGR subsystem to obtain logrec log stream error records.	z/OS MVS Programming: Assembler Services Guide See z/OS MVS Diagnosis: Tools and Service Aids for logrec specific information.
Delete logrec log data.	<ul style="list-style-type: none"> • “Managing log data: How much? For how long?” on page 272 • “Deleting log data and log data sets” on page 291
Tune the logrec log stream definition using SMF record type 88.	z/OS MVS System Management Facilities (SMF)

Table 17. Finding Logrec Log Stream information (continued)	
What is your goal?	You need to read:
Determine your current and alternate logrec medium names: <ul style="list-style-type: none"> To find your current logrec medium, issue the DISPLAY LOGREC command. To find the current and alternate logrec mediums, use the IEANTRT service. 	For the DISPLAY LOGREC command, see z/OS MVS System Commands . For the IEANTRT service in an authorized environment, see z/OS MVS Programming: Authorized Assembler Services Reference EDT-IXG . For the IEANTRT service in an unauthorized environment, see z/OS MVS Programming: Assembler Services Reference IAR-XCT .
Issue the SETLOGRC command to change your logrec medium.	z/OS MVS System Commands
At IBM's request, gather system logger component trace data.	z/OS MVS Diagnosis: Tools and Service Aids

Finding information for APPC/MVS

Use [Table 18 on page 242](#) to find information to help you set up the log stream for APPC/MVS to use during resource recovery. An APPC/MVS log stream is required for protected conversations support.

Table 18. Finding APPC/MVS information	
What is your goal?	You need to read:
Plan for and set up the APPC/MVS log stream.	“Preparing to use system logger applications” on page 243 contains most of the information you need to plan and set up a log stream for APPC/MVS. For information that is specific to APPC/MVS, however, see the topic on protected conversations support in z/OS MVS Planning: APPC/MVS Management , which contains: <ul style="list-style-type: none"> The required naming convention to use for the APPC/MVS log stream Requirements and suggestions for the associated coupling facility structure.
Plan and set up DASD space needed for APPC/MVS log data sets and staging data sets.	“Plan DASD space for system logger” on page 260
Define the APPC/MVS log and staging data sets to global resource serialization in the GRSRNLxx parmlib member.	“Add the DASD data sets to the GRSRNL inclusion list” on page 267
Define authorization to system logger resources for APPC/MVS.	“Define authorization to system logger resources” on page 273
Format the LOGR couple data set, identify it to the sysplex, and make it available.	“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276
Add, update, or delete APPC/MVS policy data (the APPC/MVS log stream and its associated coupling facility structure) in the LOGR policy using the IXCMIAPU utility.	“LOGR keywords and parameters for the administrative data utility” on page 358
Update the CFRM sysplex couple data set with the coupling facility structure associated with the APPC/MVS log stream.	“CFRM parameters for administrative data utility” on page 346
Activate the LOGR subsystem.	“Activate the LOGR subsystem” on page 291

Finding information for IMS common queue server log manager

Use [Table 19 on page 243](#) to find information to help you set up and use the IMS subsystem common queue server (CQS) log manager. Almost all of the information you will need is in the IMS library.

Table 19. Finding IMS (CQS) log manager information	
What is your goal?	You need to read:
Learn about the IMS subsystem uses the CQS log.	IMS/ESA® Common Queue Server Guide and Reference
Plan for and set up the IMS (CQS) log manager log streams.	IMS/ESA Common Queue Server Guide and Reference
Develop a naming convention for system logger resources.	“Develop a naming convention for system logger resources” on page 258
Size the coupling facility structure space needed for the logrec log stream.	“Determine the size of each coupling facility structure” on page 255
Plan and set up DASD space needed for IMS (CQS) log manager log data sets and staging data sets.	“Plan DASD space for system logger” on page 260
Define authorization to system logger resources for IMS (CQS) log manager.	“Define authorization to system logger resources” on page 273
Format the LOGR couple data set, identify it to the sysplex, and make it available.	“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276
Add, update, or delete IMS CQS log data (IMS CQS log streams and coupling facility structures) in the LOGR policy using the IXCMIAPU utility.	“LOGR keywords and parameters for the administrative data utility” on page 358 See IMS/ESA Common Queue Server Guide and Reference for details on recommendations for structure and log stream attributes.
Update the CFRM sysplex couple data set with IMS (CQS) log manager coupling facility structures.	“CFRM parameters for administrative data utility” on page 346
Activate the LOGR subsystem.	“Activate the LOGR subsystem” on page 291

Finding information for resource recovery services

[z/OS MVS Programming: Resource Recovery](#) contains complete information about resource recovery and system logger in addition to the information in [Chapter 10, “Planning for system logger applications,” on page 213.](#)

Finding information for system management facilities (SMF)

[z/OS MVS System Management Facilities \(SMF\)](#) contains complete information about SMF records and system logger in addition to the information in [Chapter 10, “Planning for system logger applications,” on page 213.](#)

Preparing to use system logger applications

Before using the system logger services or system logger applications an installation must do the following:

- [“Understand the requirements for system logger” on page 244](#)
- [“Plan the system logger configuration” on page 247](#)
- [“Determine the size of each coupling facility structure” on page 255](#)
- [“Develop a naming convention for system logger resources” on page 258](#)
- [“Plan DASD space for system logger” on page 260](#)
- [“Managing log data: How much? For how long?” on page 272](#)
- [“Define authorization to system logger resources” on page 273](#)
- [“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276](#)
- [“Add information about log streams and coupling facility structures to the LOGR policy” on page 280](#)
- [“Define the coupling facility structures attributes in the CFRM function couple data set” on page 289](#)

- “Activate the LOGR subsystem” on page 291.

Understand the requirements for system logger

These planning steps apply for both coupling facility and DASD-only log streams. Each planning step following indicates how it applies to coupling facility versus DASD-only log streams. When there is a difference in how a step should be performed based on the type of log stream, this is indicated as well. Table 20 on page 244 illustrates the steps required for setting up both types of log streams.

Common Setup Activities		Additional CF Log Stream Steps	
Sysplex Environment			
<ul style="list-style-type: none">• Need to be in a sysplex environment (PLEXCFG)• Set up sysplex, LOGR couple data sets and LOGR policy• Parmlib members — IEASYSxx, COUPLExx, GRSRNLxx, IXGCNFxx		<ul style="list-style-type: none">• Need to be in a Parallel Sysplex environment• Add coupling facility• Set up CFRM couple data sets and policy	
Log Stream Configuration			
<p>Plan</p> <ul style="list-style-type: none">• Types of log streams• How many log streams• Log stream recovery		<p>Map log streams to CF structures</p> <ul style="list-style-type: none">• Determine structure sizes• Plan structure placement• Plan peer connector recovery	
Naming Conventions			
<ul style="list-style-type: none">• Log stream names• Log stream DASD data set names• Add log stream DASD data set names to GRSRNLxx inclusion list• For a system using either couple data set type LOGRY or LOGRZ, use GRSRNLxx RNLDEF RNL(EXCL) in the IPL system parameter specification for that system		<ul style="list-style-type: none">• CF structure names	
DASD Space			
<ul style="list-style-type: none">• Allow for DASD expansion, DSEXTENTS• Plan use of retention period for DASD log stream data• Plan log stream staging data set size and access• Plan SMS data set characteristics for log stream data sets• If multisystem sysplex:<ul style="list-style-type: none">– Need shared catalog– Access to DASD volumes– Serialization mechanism– SHAREOPTIONS(3,3)		<ul style="list-style-type: none">• Select duplexing method for log stream data	
Security Authorization			
<ul style="list-style-type: none">• IXGLOGR address space• Administrator of LOGR and if applicable, LOGRY and LOGRZ, couple data set• Exploiters of log streams		<p>CFRM couple data set and structures</p> <ul style="list-style-type: none">• For IXGLOGR address space• Administrator of CFRM couple data set• Exploiters of log stream structures	

Table 20. Reviewing System Logger Setup Activities (continued)

Common Setup Activities	Additional CF Log Stream Steps
Activate SMS	
Activate ICSF (when using encrypted log stream data sets)	
Activate LOGR Subsystem	
Establish SMF88 Reporting	

Coupling facility requirement for system logger

If your installation plans to use any coupling facility log streams, system logger requires a coupling facility running coupling facility control code at one of the following levels:

- CFLEVEL 1 with service level 4.03 or higher.
- CFLEVEL 2 or higher.

This requirement applies to coupling facility log streams only. See “Determine the right kind of log stream for each application” on page 247 for determining whether you should use coupling facility versus DASD-only log streams.

If you are testing coupling facility function using the Integrated Coupling Migration Facility (ICMF), then system logger requires that ICMF be at coupling facility control code at the same level as above.

The functional capabilities and storage implications associated with each CFLEVEL may vary, see “Understanding the coupling facility level (CFLEVEL)” on page 42 to determine which CFLEVEL would be appropriate for system logger's coupling facility structure use. See “LOGR couple data set versioning — new format levels” on page 321 to understand the supported system logger functional capabilities for the different LOGR CDS format levels.

Sysplex requirement

To use system logger applications with either a coupling facility or DASD-only log stream, your sysplex must be in monoplex or multisystem mode. You must specify MONOPLEX or MULTISYSTEM on the PLEXCFG parameter in the IEASYSxx parmlib member or IPL option. System logger will not operate in XCFLOCAL mode.

See *z/OS MVS Diagnosis: Reference*, major resource name SYSZLOGR, for information related to system logger logstream resource name usage on ENQ and DEQ requests.

GRS complex considerations

If the environment in which system logger is executing consists of more than one sysplex in a GRS complex, with like-named logstreams defined on multiple sysplexes, the following requirements must be met:

- The GRS Ring complex contains at most one multisystem sysplex.
- One of the following must be true:
 - Each like-named log stream in the GRS complex must have a unique EHLQ/HLQ.
 - The installation must ensure the separation of system logger log stream resources (separate catalogs and DASD). Further, the log stream offload data set naming convention must be included in the inclusion list as discussed in “Add the DASD data sets to the GRSRNL inclusion list” on page 267.

When a single-system scope system logger couple data set type of either LOGRY or LOGRZ is used on a system in the sysplex, with like-named log streams defined in LOGRY or LOGRZ single-system scope CDS policy and in the LOGR sysplex scope policy, the following requirements must be met:

- One of the following must be true:

- Each like-named log stream between the different system logger CDS data types must have a unique EHLQ/HLQ.
- The installation must ensure the separation of system logger log stream resources (separate catalogs and DASD).

Further, on a system using the single-system scope CDS data type, use GRSRNLxx RNLDEF RNL(EXCL) in the IPL system parameter specification for that system to identify which log stream related data set allocations will be for just SYSTEM scope. Refer to [“Add the DASD data sets to the GRSRNL inclusion list”](#) on page 267.

System managed storage (SMS) requirement for system logger

For either a coupling facility or a DASD-only log stream, system logger requires that you have SMS installed and active at your installation. This is true even if you do not use SMS to manage the DASD log data sets and staging data sets, because system logger uses SMS to allocate data sets, such as the DASD log stream and staging data sets.

If SMS is not active when you attempt to use a system logger application, system logger issues system messages that indicate allocation errors.

IBM recommends that you use SMS to manage DASD data sets. Alternatively, you can establish a minimal SMS configuration to provide an environment for SMS, but no data sets become system-managed. See [“Set up the SMS environment for DASD data sets”](#) on page 266.

Integrated Cryptographic Service Facility (ICSF) requirement for system logger

When you are using or planning to use encrypted log stream data sets, system logger requires that you have ICSF installed and active at the time of your installation.

If ICSF is not active when you attempt to use a system logger application, system logger issues system messages that indicate data set allocation or open errors.

For more information, see [“Considerations for encrypting log stream data sets”](#) on page 236 and [“Installation, initialization, and customization”](#) in *z/OS Cryptographic Services ICSF System Programmer's Guide*.

Authorization requirements for system logger

To use system logger applications, installations must define authorization to system logger resources, such as log streams or coupling facility structures associated with log streams. For more about the authorization requirements for system logger, see [“Define authorization to system logger resources”](#) on page 273.

Logger resource configuration consistency requirements

The system logger environment consists of several components. The logger resource configuration data across all the components must be in a time consistent state for it to be used during a recovery situation.

A consistent view of the logger resource configuration must include:

- the active LOGR couple data set (primary or alternate) which includes:
 - policy information:
 - log stream definitions
 - coupling facility list structure definitions, if applicable
 - data containing the current state of log streams, including but not limited to the following:
 - whether a log stream is currently connected to the coupling facility structure
 - the duplex state on a system connection basis

- the directory of log stream offload data sets that house log data moved from interim to secondary log stream storage
- any active single-system scope LOGRY or LOGRZ couple data set (primary or alternate) which includes similar type information as LOGR couple data set above, without any coupling facility information.
- log stream offload data sets
- log stream staging data sets
- the catalogs (catalog structure) which maintains the location of all the log stream data sets
- the system logger coupling facility structures, if applicable

For a disaster recovery purpose, if you dump the volume with the master catalog (or user catalog) on it and then dump the volume with the system logger policy on it, then the policy might indicate that certain data sets should exist. However, those data sets might not be reflected in the catalog. If your goal is to preserve the system logger environment without losing any data, then you cannot rely on volume dumps to transport the data to the recovery site.

Since you cannot rely on volume dumps to restore the system logger environment at a recovery site, you must use some mechanism that provides for time consistent data across all the components.

Any DASD mirroring and/or coupling facility system-managed structure duplexing approaches that are employed need to ensure the logger resource configuration is consistent on any recovery site/system. Also, all the log stream exploiter's data will need to be consistent with the logger resource configuration in order for the exploiter to properly recover.

See ["System logger recovery" on page 300](#) and ["LOGR couple data set use considerations" on page 325](#) for additional considerations.

Plan the system logger configuration

When you set up your logging configuration for a system logger application, you should do some basic planning. See the following topics:

- ["Determine the number of log streams required" on page 247](#). This step is required for both coupling facility and DASD-only log streams.
- ["Determine the right kind of log stream for each application" on page 247](#).
- ["Determine which log streams map to which coupling facility structures" on page 248](#). Do this for coupling facility log streams only.
- ["Plan your configuration for log data recovery" on page 251](#). Do this for coupling facility log streams only.

Determine the number of log streams required

For both coupling facility and DASD-only log streams, you must plan the number of log streams required by system logger applications for your installation based on the kinds of logger applications you plan to have and the type of log stream, coupling facility or DASD-only. For example, one application might run on multiple systems in the sysplex and require a single sysplex-wide log stream. For another application running on multiple systems, you might want a separate log stream for each system.

For OPERLOG and logrec log streams, if you want to merge data from across the sysplex, you define one coupling facility log stream for the sysplex, written to by all the systems.

Determine the right kind of log stream for each application

Now that you know how many log streams you need and what system logger applications your installation will use, you must determine what kind of log stream is right for each application - coupling facility or DASD-only log stream. You will define the log stream as either coupling facility or DASD-only in the log stream definition in the LOGR policy in a later step in the setup process (see ["Specifying whether the log stream is DASD-only" on page 281](#)).

To decide which type of log stream is right for you, consult the application documentation or programmer for the system logger applications your installation uses and keep in mind the following:

- Does the application support DASD-only log streams? DASD-only log streams are supported for the following IBM applications:
 - CICS log manager
 - Operations log stream (OPERLOG)
 - Logrec log stream
 - Resource Recovery Services

For other system logger applications, consult the programmer or the application documentation.

- If an application needs to merge data from multiple systems, you must use a coupling facility log stream, which requires, of course, a coupling facility.
- If you do not have access to a coupling facility, a DASD-only log stream is the only kind you can use.
- DASD-only log streams can only be single system in scope. Only one system can connect to a DASD-only log stream. Multiple instances of an application may connect to a DASD-only log stream, but they must all be on the same system.

Use a DASD-only log stream when you do not have access to or do not want to use a coupling facility for logging.

- You can use a coupling facility log stream for single system scope applications, but a coupling facility is still required.

Here are some application examples:

- An application runs on multiple systems in the sysplex and requires a single sysplex-wide log stream. For this application, you would have to use a coupling facility log stream, because data is merged from multiple systems to a single log stream. This requires a coupling facility.
- Another application runs on multiple systems and requires a separate log stream for each system. For this application you can use either a DASD-only or a coupling facility log stream. However, if you use a coupling facility log stream you must have a coupling facility.

Note that an installation can consist of coupling facility log streams only, DASD-only log streams only, or a combination of both, as long as all requirements are met.

Determine which log streams map to which coupling facility structures

Each installation planning for coupling facility log streams needs to plan how to assign log streams to coupling facility structures. This can be done in a variety of ways: multiple log streams sharing one structure, one structure for each log stream, related log streams using a structure, and so forth. The important point is that the approach be systematic, easy for your installation to track, and easy to recover data for, in the event of a system, sysplex, or coupling facility failure.

When you plan how log streams will map to coupling facility structures, there are some limitations to consider, such as the maximum allowable numbers of:

- Structures you can define to a coupling facility.
- Connectors you can define to a coupling facility structure.
- Structures you can define in a CFRM policy.

For coupling facility limitations, see the *PR/SM Planning Guide*. Structure limitations are documented with the IXCL1DSU utility for the CFRM and LOGR policies. See [Chapter 11, “Format utility for couple data sets,”](#) on page 313 for the IXCL1DSU utility.

When you have more than one log stream using a single coupling facility structure, system logger divides the structure storage evenly among the log streams that have at least one connected system logger application. For example, if an installation assigns three log streams to a single structure, but only one log stream has a connected system logger application, then that one log stream can use the entire coupling facility structure. When an application connects to the second log stream, system logger dynamically

divides the structure evenly between the two log streams. When an application connects to the third log stream, system logger allocates each log stream a third of the coupling facility space. For this reason, it is important that the log streams that share a coupling facility structure be equal in size and usage rates. Knowing this can help you understand and plan how much coupling facility space you have for each log stream and predict when log stream data will be written to DASD as the coupling facility space becomes filled.

Start planning your configuration by determining the number of coupling facility structures available for logging, then determine the number of log streams needed for all system logger applications. Group log streams with the same characteristics together in coupling facility structures. **IBM recommends** that you limit the number of log streams per structure to 10 or fewer. This is because system logger connect and log stream recovery processing (which affect the restart time of system logger applications) have been optimized to provide parallelism at the coupling facility structure level. Therefore, the more structures you use, the greater parallelism you'll get during log stream connect and rebuild processing.

For example, if you have 10 log streams, each one defined to its own unique coupling facility structure, system logger can process connect requests to each log stream in parallel. On the other hand, if you define all 10 log streams to the same coupling facility structure, system logger must sequentially process log stream connect requests for the log streams.

The LOGSNUM parameter

You will define the maximum allowable number of log streams that map to a coupling facility structure on the LOGSNUM parameter for each structure in the LOGR policy in a later step, ([“Add information about log streams and coupling facility structures to the LOGR policy” on page 280](#)).

Because system logger divides the structure storage evenly among each log stream that writes to it, it is important to plan carefully what value you want to specify for LOGSNUM and how many of the log streams defined to a shared coupling facility structure will be active at any one time. The more connected log streams that share a coupling facility structure, the less coupling facility space each one has. In addition, when a structure is allocated for system logger, some of the structure space is taken up for control information for each log stream that could be connected to this structure. The higher the number for LOGSNUM, the more coupling facility control space is consumed.

In general, the value you define for LOGSNUM should be as small as possible. IBM recommends that you configure 20 or fewer log streams into one structure.

Separating system logger production and test log streams

Starting with z/OS V1R8, you can separate log streams into two groups, production and test. This means that you can separate system logger processing for test log streams from your production work log streams on a single system. Your production log streams are then protected from a hang or failure in the system logger test environment.

To separate log streams into production and test log streams you can use the GROUP(PRODUCTION | TEST) parameter in the following ways:

- To group log streams using a batch program, use the GROUP(PRODUCTION | TEST) parameter on the DEFINE or UPDATE requests on the administrative data utility, IXCMIAPU.
- To group log streams using a program, use the new GROUP(PRODUCTION | TEST) parameter on the DEFINE or UPDATE requests on the IXGINVNT service.

When you have log streams separated into production and test groups, system logger will do data set processing, such as data set allocations, data set recalls, and other functions for the two log stream groups in two different sets of tasks. In addition, system logger limits resource consumption of TEST log streams as follows:

- TEST log streams are limited to using a maximum of 25% of the connections allowed, while PRODUCTION log streams can use at least 75% of connection slots.
- TEST log streams are limited to using a maximum of 25% of LOGR, LOGRY or LOGRZ couple data set extents allowed, while PRODUCTION log streams can use at least 75%.

By default, log streams are PRODUCTION log streams. This means that existing log streams with no GROUP designation are PRODUCTION log streams.

Using structures with grouped log streams: A coupling facility structure can only have one type of log stream assigned to it, either TEST or PRODUCTION. If you try to assign a TEST log stream, for example, to a STRUCTURE with PRODUCTION log streams, the request will fail with a reason code of IxgRsnCodeBadGroup (X'08E9').

The first log stream that is defined to a structure determines what type of log streams can be defined to that structure. If the first log stream defined to a structure is a TEST log stream, you can only define TEST log streams to that structure. If you specify or default to PRODUCTION for the first log stream defined to a structure, you can only define other PRODUCTION log streams to that structure.

Other support for separating test and production log streams: The following interfaces support the separation of test and production log streams:

- The DISPLAY LOGGER command displays the group designation for log streams. See [Displaying the system logger and its log streams in z/OS MVS System Commands](#).
- The IXCMIAPU utility LIST LOGSTREAM request displays the group designation for log streams. See “LIST LOGSTREAM keywords and parameters” on page 384.
- SMF record type 88, subtype 1, Log Stream section, includes field SMF88GRP to display the group designation for each log stream. See [Log stream section in z/OS MVS System Management Facilities \(SMF\)](#).
- IXGQUERY will return the log stream group designation as long as you specify a large enough buffer length (200 bytes or greater). See [IXGQUERY - Query a log stream for information in z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).
- ENF signal 48 for DEFINE and UPDATE log stream requests will identify the group of the log stream.

Examples of coupling facility log stream configurations

The following examples show different ways to associate log streams and coupling facility structures for system logger applications. There are many other ways to set up a configuration for your system logger application.

Example - merging log data from across the sysplex

The scenario in [Figure 63 on page 250](#) shows a configuration where a transaction manager system logger application runs on two systems in a sysplex. The applications write and merge log data into one log stream, named TRANSLOG in this example. This configuration is the one you would set up for the OPERLOG or logrec log stream applications.

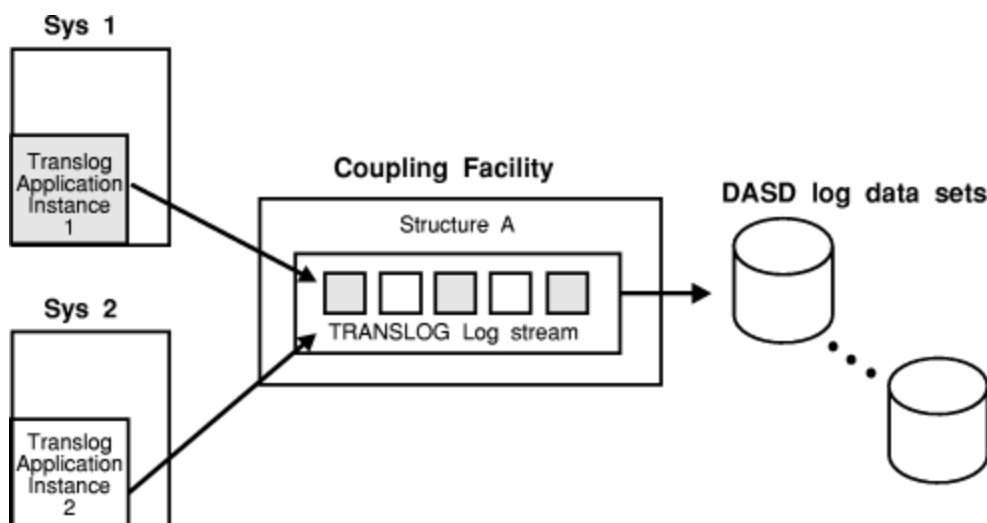


Figure 63. Example: Merging Log Data from Systems Across a Sysplex

Some of the key points this example shows are:

- Log data is written to the TRANSLOG log stream on a coupling facility list structure.
- Multiple instances of a system logger application running on different systems to share data.
- The TRANSLOG log stream is the only log stream occupying the coupling facility list structure.
- System logger automatically moves the oldest log data from the coupling facility to DASD when the data hits the installation-defined thresholds for the coupling facility list structure for this log stream. From the point of view of the system logger application, it does not matter whether the log data resides on the coupling facility structure or on DASD.

Example - maintaining multiple log streams in the same coupling facility structure

Figure 64 on page 251 shows a configuration where you can maintain individual logs for each system logger application in separate, independent log streams that map to the same coupling facility list structure. You might, for example, want to have your logrec log stream and OPERLOG log stream map to the same coupling facility structure.

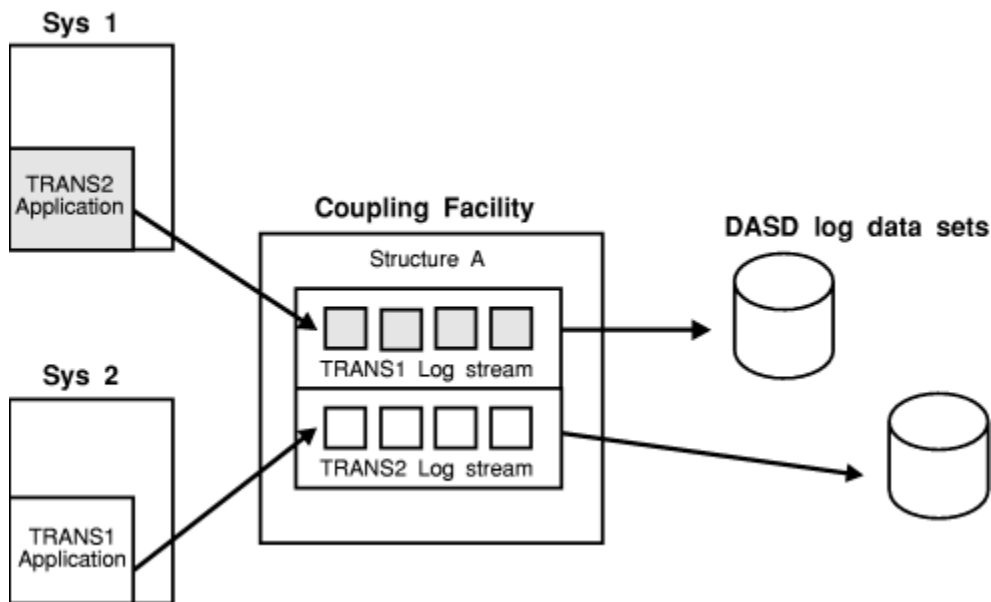


Figure 64. Example: Maintaining Multiple Log Streams in the Same Coupling Facility Structure

Some of the key points this example shows are:

- Although they reside in the same coupling facility structure, the two log streams are separate. The data from the two log streams is **not** merged.
- System logger automatically moves the oldest log data from the coupling facility to DASD when the data hits the installation-defined thresholds for the coupling facility list structure for a log stream.

Plan your configuration for log data recovery

When you are deciding which coupling facility structures to associate with each coupling facility log stream, you can plan it so that system logger can easily recover coupling facility data in the case of a system failure. This step applies to coupling facility log streams only.

When a system that has a system logger application fails, system logger tries to safeguard all the coupling facility log data for the failed system either by offloading it to DASD log data sets so that it is on a persistent media or by ensuring that the log data is secure in a persistent duplex-mode list structure. This recovery processing is done by a **peer connector**, which is another system in the sysplex with a connection to the same coupling facility structure as the failing system. In general, when you set up your logging configuration, make sure that each log stream structure is accessed by more than one system. Note that a peer connector need only be connected to the same coupling facility structure, not the same log stream.

If there is no peer connection available to perform recovery for a failed system, recovery is delayed until either the failing system re-IPLs or another system connects to a log stream in the same coupling facility structure to which the failing system was connected. In general, however, the log stream data you need will be available when you need it: when a system reconnects or connects to a log stream that the failed system was connected to in order to write or retrieve data. Recovery will be performed when that peer connector connects to the log stream.

Examples - peer connectors recovering coupling facility log data for a failing system

In Figure 65 on page 252, system SYS1 fails. System logger recovery processing for SYS1 consists of moving all the coupling facility resident log data for the system to DASD log data sets. Because SYS1 is connected to log streams on coupling facility structure A, it needs a peer connector with a connection to structure A to perform recovery. SYS2 can perform recovery for SYS1 because it has a system logger application connected to a log stream that maps to coupling facility structure A. SYS3 is not a peer connector for SYS1 and cannot perform recovery because it does not have a connection to coupling facility structure A.

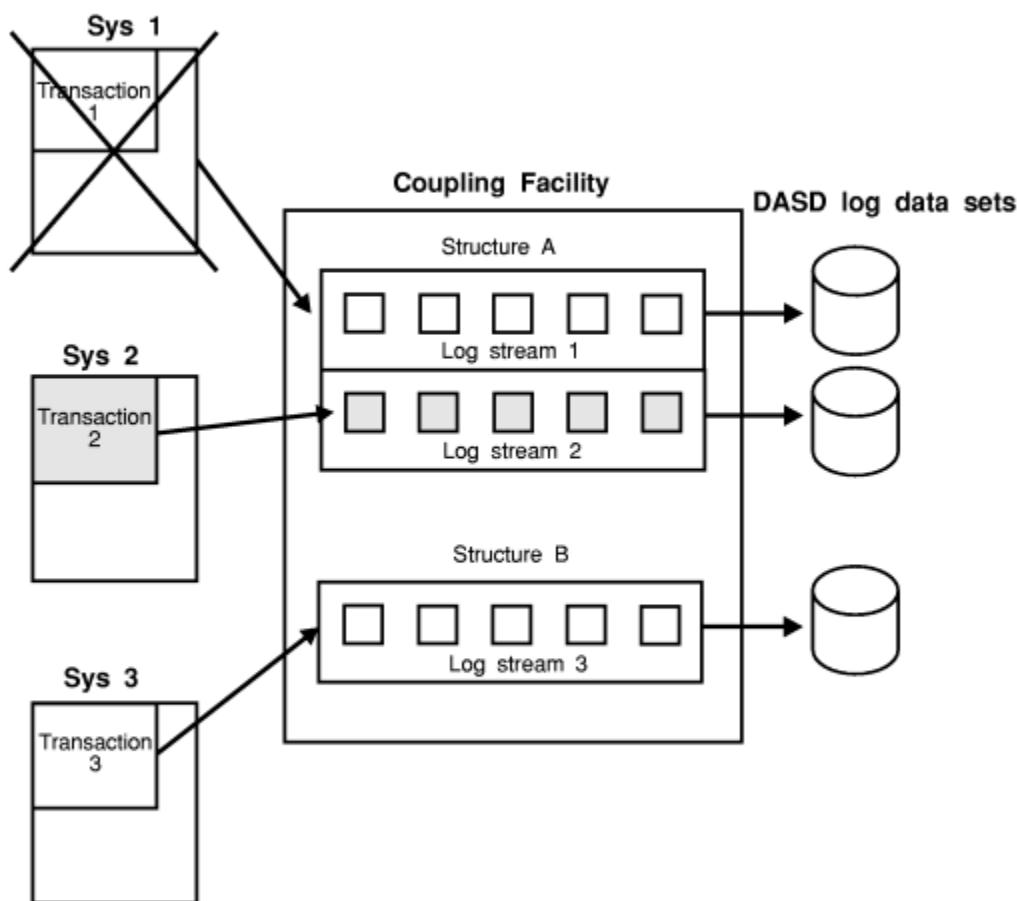


Figure 65. Example: Recovery for a Failing System - All Data Recovered

In Figure 66 on page 253, system SYS1 also fails. SYS1 has connections to coupling facilities structures A and B and so needs peer connectors to both structures to recover all its coupling facility resident log data. SYS2 can perform recovery for SYS1 log data on coupling facility structure B, but there is no peer connector to perform recovery for SYS1's log data on coupling facility structure A.

Recovery for SYS1's coupling facility structure A log data is delayed until either SYS1 re-IPLs or another system connects to a log stream on coupling facility structure A.

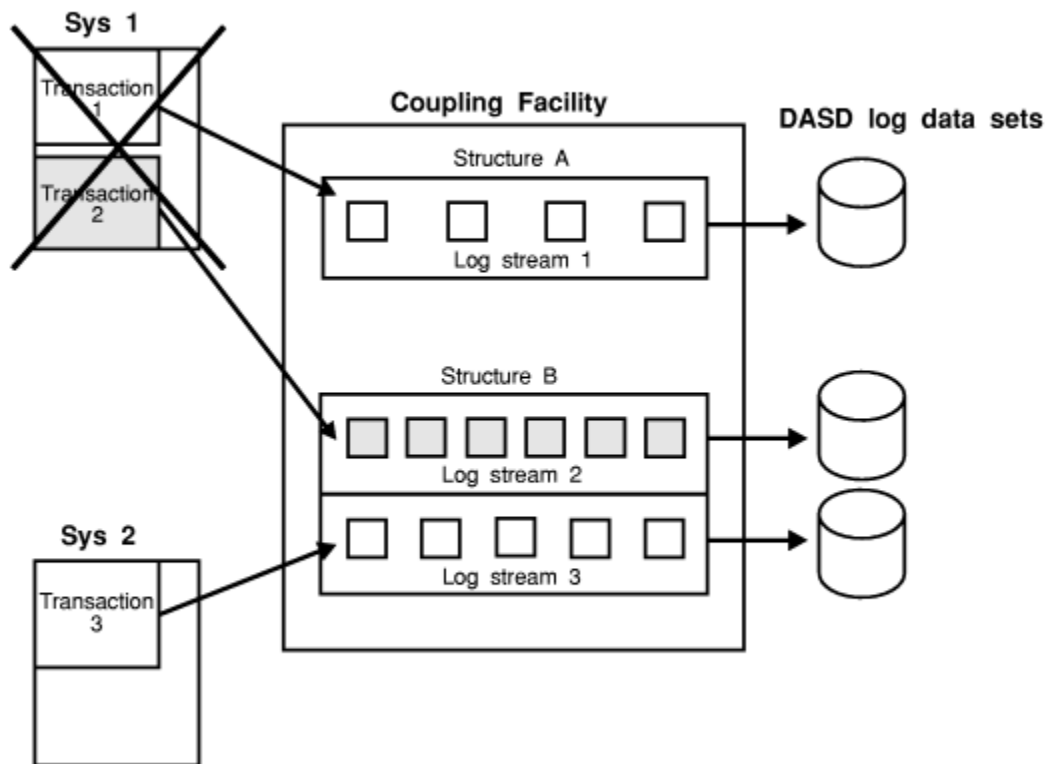


Figure 66. Example: Recovery for a Failing System - Recovery Delayed for Some Data

Managing log data in interim storage to minimize movement to DASD

You can use a very large coupling facility structure in the magnitude of gigabytes vs. megabytes to minimize movement to DASD for cases where log data deletion is actively managed by the exploiter. The use of a supersized CF structure is intended to minimize the likelihood of log stream offload delays that impact an exploiter that writes log data into the log stream. Coupled with configuration changes and taking advantage of monitoring and automation, this will allow the installation a path a greater resiliency and minimize the impact of offload inhibitors when they occur. Offload inhibitors generally fall into two categories:

DASD allocation hang

happens when logger data set allocation hang obtaining "next to be used as current" offload data set.

DASD I/O bandwidth slowdown

happens when an I/O performance delay causes the rate of a new log data written into a log stream to exceed system logger's ability to write log data to offload data sets and subsequently make more room in the structure.

The offload inhibitors often occur for reasons outside of the control of system logger. Due to the nature of these problems, the supersized structure environment takes the approach of configuring the log stream and its exploiter in a manner so that the log stream does not need to be physically offloaded to DASD during normal activities. This will assist in avoiding potential problems with offload dataset allocation or I/O performance. This approach is already taken by CICS Transaction Server (CTS) with respect to configuring a CICS system log. The same type of approach can be used by other log stream exploiters.

The installation will need to make operational changes to the specifications on log stream and CF structure attributes to provide the installation awareness and ability to react if the log stream primary/interim resource is becoming constrained. The installation should focus on taking the necessary action to allow a log stream exploiter to logically delete the data periodically.

When sizing and setting these attributes, you should keep in mind that the goal is to allow a "warning track" for log stream offload processing in case an inhibitor is encountered. The installation will set a low percentage for the logger HIGHOFFLOAD so that normally very little of the CF structure would be used. If the offload is keeping up with the write rate, the HIGHOFFLOAD threshold should not be significantly exceeded.

The CF structure has been changed to allow for a more automated approach. System logger provides the user with programmatic data about structure usage. IMS Common Queue Services (CQS) will act on this data to determine when to checkpoint. If you use IMS CQS as an example of a log stream exploiter that is usually considered a funnel-type log stream user with the following algorithm, you can use an active-type log stream user. These changes are only effective when the log stream and related resources are configured correctly. For specific references on CTS and IMS CQS using log streams, see ["Finding information for CICS log manager" on page 239](#) and ["Finding information for IMS common queue server log manager" on page 242](#).

The algorithm used for determining the various CFRM specifications is as follows:

Structure Size

The structure size = 2 IMS check points of data + safety zone + 2 minutes of data + reaction time data.

- 2 IMS check points of data = 2 * amount of data written in one minute * the number of minutes between structure checkpoints.
- Safety Zone = 20% of the 2 IMS check points of data.
- Reaction time data = (the minutes of desired reaction time + 1) * the data written in one minute.

Ensures that the structure is large enough to hold 2 IMS CQS check points of data and create a reaction time or warning track to handle any offload issues if system logger needs to physically offload data.

INITSIZE and MINSIZE

Allows XES to manage the CF structure size based on the above SIZE specification.

Note: Do not code these options.

FULLTHRESHOLD

$$\text{FULLTHRESHOLD} = (100 - \text{HIGHOFFLOAD}) / 3 + \text{HIGHOFFLOAD}$$

The IXC585E is the warning message for bandwidth issues causing offload delays. Ensure the IXC585E comes out early enough in an error situation to permit reaction and ensure the message is not issued frequently during normal operation.

ALLOWAUTOALT(NO)

CF structures used for log streams generally have ALLOWAUTOALT(NO) specification.

The considerations for using this option are explained in ["Define the coupling facility structures attributes in the CFRM function couple data set" on page 289](#).

The algorithm used for determining the various LOGR specifications is as follows:

STG_DUPLEX(NO) |

STG_DUPLEX(YES) and DUPLEXMODE(UNCOND|COND)

The previous restriction stipulated that the super-sized structure works only if the staging dataset was not being used.

A log stream staging data set can be sized to match the CF structure size, even when a supersized structure is being used.

RETPD

RETPD = 0 (default)

A retention period (RETPD) value forces log data to be offloaded to DASD before it can be deleted. Specifying a nonzero retention period value defeats the purpose of sizing a CF structure to large to avoid offloading during peak times.

If the installation must use the RETPD value to force data to be retained for a period of time, then you should not use this super-sized structure approach as a mechanism to avoid offloads.

HIGHOFFLOAD

$\text{HIGHOFFLOAD} = (2 \text{ IMS check points of data} + \text{safety zone} + 2 \text{ minutes of data}) / \text{total structure size}$

Ensure the LOWOFFLOAD and HIGHOFFLOAD are sufficiently close together so that the offload happens more frequently. Ensure the LOWOFFLOAD and HIGHOFFLOAD are far enough apart so that system logger is not continuously offloading.

LOWOFFLOAD

$\text{LOWOFFLOAD} = (2 \text{ IMS check points of data} + \text{safety zone}) / \text{total structure size}$

Ensure the LOWOFFLOAD is set so that 2 IMS CQS checkpoints of data remain in the structure.

LS_DATACLAS

$\text{LS_DATACLAS} = 24 \text{ CI size and striping.}$

Ensure making use of recommended and higher performing attributes for offload data sets.

LS_STORCLAS

$\text{LS_STORCLAS} = \text{striping.}$

Ensure making use of recommended and higher performing attributes for offload data sets.

Although this approach is designed to minimize the offloading of data to DASD, it must be noted that no matter how large a log stream's primary storage is configured, system logger will write data to offload data sets for certain events or activities. When the connector of the IMS CQS log stream terminates on a given system or if there is a major spike in the workload, a rebuild will occur.

CF structure resident log data can be duplexed by XES structure duplexing, system logger local buffers, or some combination of these. The factors that determine this are the definition of the log stream and structure resources, as well as the environment of the system. If the intent is to use a duplexed CF structure with XES system managed duplexing, then the installation will need to be aware that if the structure falls back to simplex mode, system logger will initialize an offload to move all eligible structure resident log data to DASD. System logger will then have control and the component has to ensure log data is solidified when it occurs. If the intent is to use an environment where local buffers are the duplexing medium, the installation must ensure it has adequate page data set space available to back up the local buffers it needs.

This supersized structure raises considerations with regards to certain recovery cases. One such case is the existing system logger processing with regards to the loss of a connected CF structure. Currently, certain structure failures will trigger system logger to allocate a staging data set when a rebuild to a new structure fails. In this case, system logger will allocate a staging data set to harden the log data so it can be recovered later. This is done to preserve the data until a structure can be allocated. If the installation has not defined the staging data set parameters that control the size, the allocate will fail since the default system logger behavior is to use the CF structure size when sizing the staging data set. If the installation defines the staging data set related log stream parameters, then system logger may be able to preserve some of the data. This staging data set fallback copy is important as it is a method of preserving the log data against cases where a system logger image fails since the local buffer copies of the data would be lost when the address space goes away.

Determine the size of each coupling facility structure

If you are using coupling facility structures for your log streams, you must specify in the CFRM policy the size of each structure you define. To simplify the task of calculating the size, IBM provides the CF Structure Sizer (CFSizer), a tool that estimates the storage size for each structure by asking questions based on your existing or planned configuration. The CF Structure Sizer (CFSizer) uses the selected structure input to calculate the SIZE and INITSIZE values for the CFRM policy.

The IBM CF Structure Sizer (CFSizer) can be accessed at [Coupling Facility sizer \(www.ibm.com/support/docview.wss?uid=isg3T1027062\)](http://www.ibm.com/support/docview.wss?uid=isg3T1027062).

Before using the CF Structure Sizer (CFSizer), you should go through the following steps for each coupling facility structure you wish to define. The information that you collect will be useful both for the CF Structure Sizer (CFSizer) and for defining your LOGR policy. The procedure will help you determine the minimum amount of coupling facility space you should allocate for a coupling facility structure, based on the amount of space needed for all the log streams mapping to the structure. See [“Add information about log streams and coupling facility structures to the LOGR policy”](#) on page 280.

1. Gather information about the coupling facility structure.

For each structure, gather the following information common for all the log streams that map to the structure.

AVGBUFSIZE

Average size of log blocks written by applications to log streams associated with this coupling facility structure. This value corresponds to the AVGBUFSIZE parameter specified for each structure in the LOGR policy. For this procedure, pick the value that represents the average size of log blocks generated by all the log streams writing to this coupling facility structure. For specifying the AVGBUFSIZE parameter in the LOGR policy, see [“Specifying the average log block size”](#) on page 281.

For a logrec log stream, **IBM recommends** an AVGBUFSIZE of 4068.

For an OPERLOG log stream, **IBM recommends** an AVGBUFSIZE of 512.

For an RRS log stream, see [z/OS MVS Programming: Resource Recovery](#).

Installation or vendor supplied applications should base AVGBUFSIZE on the average size of the log blocks they want to write to the log stream.

MAXBUFSIZE

Maximum size of a log block corresponding to the MAXBUFSIZE parameter in the LOGR policy.

For a logrec or OPERLOG log stream, **IBM recommends** a MAXBUFSIZE of 4096. For an RRS log stream, see [z/OS MVS Programming: Resource Recovery](#).

Installation or vendor supplied applications should base MAXBUFSIZE on the maximum size of the log blocks they want to write to the log stream.

Note that once you have defined the MAXBUFSIZE for a coupling facility structure, you cannot update the value. To change the MAXBUFSIZE, you must delete the log streams associated with the structure, delete the structure, and then re-define the structure with the new MAXGBUFSIZE value.

LOGSNUM

Maximum number of log streams in the structure corresponding to the LOGSNUM parameter in the LOGR policy. See [“The LOGSNUM parameter”](#) on page 249 for more information.

2. Gather information about each log stream

For each log stream that maps to a coupling facility structure, gather the following information.

Most of these values are projected or desired values for your log stream, and a few correspond to parameters you will specify in the LOGR policy.

LOWOFFLOAD

LOWOFFLOAD specifies the point, in percent value of space consumed, where system logger will stop offloading coupling facility log data to the DASD log data sets for this log stream. It corresponds to the LOWOFFLOAD parameter in the LOGR policy. For OPERLOG, logrec log stream, or the CICS log manager, **IBM recommends** that you use the LOWOFFLOAD default value of zero. See [Chapter 12, “Administrative data utility,”](#) on page 335 for more the LOWOFFLOAD parameter.

HIGHOFFLOAD

HIGHOFFLOAD specifies the point, in percent value of space consumed, where system logger will begin offloading coupling facility log data to the DASD log data sets for this log stream. It corresponds to the HIGHOFFLOAD parameter in the LOGR policy.

For OPERLOG, logrec log stream, or the CICS log manager, **IBM recommends** that you use the HIGHOFFLOAD default value of 80. See [Chapter 12, “Administrative data utility,” on page 335](#) for more the HIGHOFFLOAD parameter.

ResidencyTime

Desired residency time for log data in the coupling facility structure, in seconds. Residency time is the amount of time you want a log block to stay in the coupling facility between the time it is written to the coupling facility and being off-loaded to DASD.

For a logrec or OPERLOG log stream, **IBM recommends** a residency of 10 seconds. For other system logger applications, get the desired residency time from the documentation for the application.

WritePerSec

Projected write requests per second against the log stream. This value is the total IXGWRITE requests per second issued by all system logger applications or systems connected to a log stream.

For an OPERLOG log stream, you can calculate projected writes per second using the current SYSLOGs. For each system in the sysplex, do the following calculation:

- a. Choose a spot in SYSLOG and note the timestamp at that spot. You might want to choose a peak or high usage time of day.
- b. Page down 2000 lines by issuing, for example, DOWN 2000 to SDSF or TSO/E browse.
- c. Note the time stamp in this spot.
- d. Calculate the number of seconds between the two time stamps. This is the number of seconds it took to write 2000 lines of SYSLOG. (This calculation is based on most messages being one line).
- e. Divide 2000 by the number of seconds it took to write 2000 lines to get the lines per second.
- f. Add the results for each system's SYSLOG to get the total writes per second for the OPERLOG log stream.

For a logrec log stream, calculate the projected writes per second using the current logrec data sets. For each system that will write to the logrec log stream, do the following calculation:

- a. Request or obtain an EREP report for a particular time span. It must be a report that includes all records. For example, take an EREP daily report that processed all records.
- b. Near the top of the report, message IFC120I will tell how many records were written in the time span chosen for the report. Divide this number by the number of seconds in the time span for the average writes per second for this logrec data set.
- c. You can also look at timestamps in the output to analyze how many records were written in a particular second. You can do this to check for peak usage.
- d. Add the results for each logrec data set that will write to the log stream to get the total writes per second for a logrec log stream.

After generating the estimated structure sizes from the CF Structure Sizer (CFSizer), if you need to change the coupling facility structure size later, there are two ways you can do that:

- Dynamically alter the coupling facility structure size by entering the SETXCF START,ALTER command or issuing the IXLALTER service.
- Update the CFRM policy with the new coupling facility size, activate the policy, and then have the operator initiate a rebuild of the structure. See [Chapter 12, “Administrative data utility,” on page 335](#) for updating the CFRM policy using the IXCMIAPU utility.

Note, however, that these options should not be substituted for advance planning, which can optimize both capacity and performance. See the following for more information:

- [z/OS MVS System Commands](#) for the SETXCF command.
- [z/OS MVS Programming: Sysplex Services Reference](#) for reference information about the IXLALTER service.

- [z/OS MVS Programming: Sysplex Services Guide](#) for guidance on using the IXLALTER service.

Develop a naming convention for system logger resources

If you are going to have multiple system logger applications, you will need to define sysplex-wide consistent naming conventions for coupling facility structures, DASD log data sets, and DASD staging data sets in order to conveniently track them. These names will subsequently be specified in the LOGR, LOGRY or LOGRZ policy.

Naming conventions for the log stream and DASD data sets

For each coupling facility and DASD-only log stream you plan, do the following:

1. Determine the log stream name.

A log stream name should be a unique descriptive identifier, made up of one or more qualifiers (each 1 to 8 characters in length) separated by periods, up to the maximum length of 26 characters. **IBM recommends** that the log stream name match the installation's data set naming conventions, with user ID or system logger application name as the first qualifier. This will simplify reporting, defining RACF generic profiles, and writing SMS ACS routines.

It is also recommended that, when possible, one qualifier in the log stream name should be variable, to distinguish between instances used by the application writing to the log stream. To use this naming method effectively, specify a different variable name to define a different instance of a log stream for use by the application. For example, a valid log stream name for one instance of a system logger application named LOG1 might be LOG1.*instance1*.RECOVERY. Another instance of that system logger application may have a log stream named LOG1.*instance2*.RECOVERY. Make sure that each instance of the application has a mapping of which log stream resource name it should be using. The variable names and the log streams they represent need to be provided to the application as part of the application's initialization or setup.

Use the following rules in naming a log stream:

- Each qualifier can contain up to eight numeric, alphabetic, or national (\$, #, or @) characters.
- The first character of each qualifier must be an alphabetic or national character.
- Each qualifier must be separated by periods, which you must count as characters.

See “[Sysplex requirement](#)” on page 245 for information related to logstream names in environments with multiple sysplexes in a GRS Ring complex.

2. Determine the high-level qualifier for log stream data sets.

Define either a high-level qualifier (*hlq*) or an extended high-level qualifier (*ehlq*). The high-level qualifier will be used in the DASD log data sets and DASD staging data sets for each log stream.

For example, depending on the needs of your installation, you could set the high-level qualifier to one of the following:

- Sysplex name
- Subsystem group name
- System logger application type name

The extended high-level qualifier is similar to the high-level qualifier, but it provides more flexibility to help meet the installation's naming conventions for log stream data sets. The active primary LOGR couple data set must be formatted at a z/OS Release 1.2 level in order to specify an extended high-level qualifier. The maximum length of the extended high-level qualifier, including periods, is 33 characters. However, the overall log stream data set names cannot be greater than 44 characters.

The high-level qualifier and the extended high-level qualifier are mutually exclusive and cannot be specified for the same log stream definition. When the *ehlq* is not specified, the resulting high level qualifier for the log stream data sets is based on whether *hlq* or the LIKE parameter is specified. If the HLQ parameter is specified, then that value will be used for the log stream data sets. When no high level qualifier is explicitly specified, but the LIKE parameter is specified, then the high level

qualifier value being used in the referenced log stream will be used for the newly defined log stream. If the extended high-level qualifier, high-level qualifier, and LIKE parameters are not specified, then the default value IXGLOGR is used.

System logger does no SAF authorization checking for the high level qualifier you select for the DASD log data sets. This means that while you can select any high level qualifier you like, you should also plan carefully the name you choose. For example, system logger will allow you to choose SYS1 as your high level qualifier, but your DASD log stream data sets will end up in your master catalog! **IBM recommends** that you plan your high level qualifier carefully and add an alias for each high level qualifier in the master catalog that points to a user catalog.

System logger uses the log stream name to generate the DASD log data sets and staging data sets:

- **To form the log data set name**, system logger combines the log stream name and the high-level or extended high-level qualifier, adding an automatically generated sequence number to form the log data set name. This combination of high level qualifier and log stream name must be unique within the sysplex.

The sequence number generated by the system consists of an 8-character string of numbers, letters, or both. You cannot try to predict or depend on the format in locating log stream data sets.

The log data set name is the name you use with SMS to manage the log data sets on DASD.

The following is an example of the log data set that is allocated when log stream name LOG1.AFORWARD.RECOVERY is defined without a high level qualifier:

```
IXGLOGR.LOG1.AFORWARD.RECOVERY.A0000001
```

Note that system logger has added the default high-level qualifier, IXGLOGR.

The following is an example of a log data set that is allocated when log stream name LOG1.AFORWARD.RECOVERY is defined with a high-level qualifier of PROD.

```
PROD.LOG1.AFORWARD.RECOVERY.A0000002
```

The following is an example of a log data set that is allocated when log stream name LOG1.AFORWARD.RECOVERY is defined with an extended high-level qualifier of MY.PREFIX.

```
MY.PREFIX.LOG1.AFORWARD.RECOVERY.A0000002
```

- **To form the staging data set name** for a connection, system logger does the following:

For a **coupling facility log stream**, system logger combines the high-level or extended high-level qualifier defined for the log stream plus the log stream name and the **system name** derived from the IEASYSxx parmlib member. See [z/OS MVS Initialization and Tuning Reference](#) for a description of IEASYSxx.

For example, a staging data set defined on system MVS1 with high level qualifier IXGLOGR and associated with log stream LOG1.AFORWARD.RECOVERY will have the following name:

```
IXGLOGR.LOG1.AFORWARD.RECOVERY.MVS1
```

System logger cannot use a system name that starts with a digit as a qualifier in a staging data set name. Therefore, if your system name begins with a digit, system logger uses 'STG' plus the last 5 significant characters of the system name to form the low level qualifier for the staging data set name. If your system name begins with a digit, you must **make sure** that the last five significant characters will identify the system uniquely within the sysplex. The following examples show how system logger converts a system name starting with a digit to a low level qualifier for a staging data set name:

- System name 1SYS starts with a digit and has less than five characters. System logger converts the digit to 'STG' and adds the four characters to come up with STG1SYS for the low level qualifier for the staging data set.

- System name 1SYSTEM starts with a digit and has more than five characters. System logger converts the digit to 'STG' and uses the last five significant characters to come up with a staging data set low level qualifier of STGSYSTEM.
- Systems 1SYSTEM and 2SYSTEM both start with a digit and have more than 5 characters. Unfortunately, this results in identical low level qualifiers of STGSYSTEM for both system's staging data set names. You might need to change the system name specified on IEASYSxx.

For a **DASD-only log stream**, system logger combines the high-level or extended high-level qualifier defined for the log stream, plus the log stream name and the **sysplex name** taken from the COUPLExx parmlib member. See [z/OS MVS Initialization and Tuning Reference](#) for a description of COUPLExx.

For example, a staging data set defined on system MVS1 in sysplex PLEX1 with high level qualifier IXGLOGR, and associated with log stream LOG1.AFORWARD,RECOVERY will have the following name:

```
IXGLOGR.LOG1.AFORWARD.RECOVERY.PLEX1
```

System logger cannot use a sysplex name that starts with a digit as a qualifier in a staging data set name. Therefore, if the sysplex name specified in the COUPLExx parmlib member begins with a digit, system logger uses 'STG' plus the last 5 significant characters of the sysplex name to form the low level qualifier for the staging data set name. If your sysplex name begins with a digit, you must **make sure** that the last five significant characters will identify the sysplex uniquely. The following examples show how system logger converts a sysplex name starting with a digit to a low level qualifier for a staging data set name:

- Sysplex name 1PLX starts with a digit and has less than five characters. System logger converts the digit to 'STG' and adds the four characters to come up with STG1PLX for the low level qualifier for the staging data set.
- Sysplex name 1SYSPLEX starts with a digit and has more than five characters. System logger converts the digit to 'STG' and uses the last five significant characters to come up with a staging data set low level qualifier of STGSPLEX.
- Sysplexes 1SYSPLEX and 2SYSPLEX both start with a digit and have more than 5 characters. Unfortunately, this results in identical low level qualifiers of STGSPLEX for both systems' staging data set names. You might need to change the sysplex name specified on COUPLExx.

When a log stream resource is renamed, it will affect the names of all the original log stream log data sets and staging data sets. See the NEWSTREAMNAME keyword in “UPDATE LOGSTREAM keywords and parameters” on page 371 of Chapter 12, “Administrative data utility,” on page 335 for a description of how the log stream (offload) and staging data sets are handled when a log stream is renamed.

Naming conventions for system logger coupling facility structures

The coupling facility structures associated with coupling facility log streams must follow structure naming conventions. The name must be:

- 1 - 16 characters long.
- Uppercase alphabetic characters (A-Z), numeric characters (0-9), national characters (\$,@,#), or an underscore (_).
- Started with an alphabetic character (A-Z).

IBM suggests that you define your coupling facility structure name to match the name of the LPAR in which the coupling facility is to run.

See Chapter 12, “Administrative data utility,” on page 335 for specifying the name in the CFRM policy using the IXCMIAPU utility.

Plan DASD space for system logger

Both coupling facility and DASD-only log streams use DASD space for log data sets and staging data sets. All of the planning steps in this topic must be performed for either coupling facility or DASD-only log streams, unless otherwise noted.

System logger allocates VSAM linear data sets for the DASD log data sets and DASD staging data sets.

IBM strongly recommends that these DASD data sets be SMS-managed.

System logger does not support the Guaranteed Space Option in SMS Storage Classes. Use of this option will cause system logger to only write to the first extent of any data set allocated in this manner.

System logger does allow VSAM data striping for DASD log data sets and DASD staging data sets.

Do not use a JCL data set retention period or expiration date to manage log data sets. See [“Managing log data: How much? For how long?”](#) on page 272 for managing your log data sets.

This topic includes the following steps for preparing your DASD log data sets and staging data sets:

- [“Plan space for DASD log data sets”](#) on page 261
 - [“Increasing the space available for DASD log data sets”](#) on page 262
- [“Plan space for staging data sets”](#) on page 263
 - [“Plan staging data sets for coupling facility log streams”](#) on page 263
 - [“Plan staging data sets for DASD-only log streams”](#) on page 265
 - [“Monitoring staging data set usage log streams”](#) on page 266
- [“Set up the SMS environment for DASD data sets”](#) on page 266
- [“Define share options for DASD data sets”](#) on page 267
- [“Add the DASD data sets to the GRSRNL inclusion list”](#) on page 267
- [“Managing logger log stream data sets”](#) on page 268
 - [“Log stream data set migration and recall considerations”](#) on page 268

Plan space for DASD log data sets

You can specify the size for DASD log data sets for each log stream on the LS_SIZE parameter in the log stream definition in the LOGR, LOGRY or LOGRZ policy.

When specified, the LS_SIZE value is used for allocation request for log stream data set. The value overrides the space allocation attributes for the data class, which can either be explicitly specified in the LS_DATACLAS parameter, or assigned by the DFSMS Automatic Class Selection (ACS) routines. The actual data set size allocated depends on factors such as track size, CISIZE, volume type, and so on.

- The smallest size that a log stream data set can be defined for system logger use is 64 KB. If the data set size is too small, logger automatically attempts to reallocate the size large enough for use.
- The largest size that a log stream data set can be defined for logger use is up to 4GB.
- If trying to achieve a data set near 4 GB in size, see [“Testing log data set parameter modifications”](#) on page 269 for important notes on choosing and testing data set size changes.

IBM recommends that you size the data sets as large as your installation can afford to make them. This will minimize the number of log data sets required to represent a log stream. It will also minimize the number of times that system logger must reallocate and switch to using a new log data set when an old one becomes full. Because allocating and switching to a new log data set incurs overhead, it should be done as little as possible.

IBM recommends that you size offload data sets no smaller than 1 MB, unless the exploiter specifically recommends a size below this amount.

For logrec log stream and OPERLOG, use your current logrec and SYSLOG output data sets as a guide for log data set size. To calculate the log data set size for logrec, add together the logrec data set sizes for all the systems writing to the logrec log stream. For OPERLOG, add the sizes of the SYSLOG data sets for each system in the sysplex and multiply by at least 2, and then use that value as your log data set size. OPERLOG log data sets will generally be at least two times greater than SYSLOG data sets, but the OPERLOG data set size may need to be altered further based on individual experience.

By default, each log stream is limited to a maximum of 168 log data sets unless you define data set directory extent records in the LOGR, LOGRY or LOGRZ couple data set and make it the active primary couple data set. By defining data set directory extent records, a log stream is no longer limited to 168 log data sets.

Increasing the space available for DASD log data sets

You can, if necessary, increase the number of log data sets available for log streams using the DSEXTENT parameter in the LOGR, LOGRY or LOGRZ couple data set. If you have log streams that will exceed 168 DASD log data sets, perhaps because you retain data in your log stream for a long period of time, increase the number of directory extents available for the sysplex on the DSEXTENT parameter in the LOGR couple data set using the format utility (see [Chapter 11, “Format utility for couple data sets,” on page 313](#)). Each additional directory extent specified goes into a common pool available to any log stream in the sysplex. System logger allocates these directory extents as needed. Each directory extent allows a log stream to extend beyond 168 log data sets. Whenever all the log data sets in a data set directory extent have been physically deleted, system logger returns the directory extent record to the available pool of directory extents for the sysplex. Each log stream has one data set directory extent which is part of the log stream record, not part of the data set directory extent pool. This permanent directory extent is always used first for a log stream, before retrieving a directory extent from the common data set directory pool. This permanent extent does not revert to the common pool when unused and is not available for use by any other log stream - it is held for the owning log stream.

To specify additional extents, format an appropriate logger CDS type (for example, LOGR) couple data set with additional directory extents on the DSEXTENT parameter. Then make that new (for example, LOGR) couple data set available as the active primary (see [“Format the sysplex scope LOGR couple data set and make it available to the sysplex” on page 276](#)).

If you decide to specify DSEXTENT, specify a moderate number of additional directory extents, at least 15% more than should be in use at any given time, and then monitor usage by:

- Periodically check your syslog or OPERLOG for system logger messages IXG320I, IXG321I, IXG322I and IXG323I. These messages indicate the total number of DSEXTENT records defined and the number in use.
- Periodically running a LOGR, LOGRY or LOGRZ couple data set report using the IXCMIAPU utility. This will show the number of directory extent records in use. Try to keep the number of directory extent records in use below 85% of the total formatted.
- Watching for system messages IXG261E and IXG262A, which indicate that usage of log data set directory extents is over 85% and 95% respectively. Watch also for message IXG301I with a return code of X'08' and a reason code of X'085C'. This indicates that an offload has failed due to a log data set directory full condition. If an offload fails because of a lack of directory extents, system logger will retry the offload after additional directory extents have been made available for the sysplex. When the offload completes and the log stream is again available, system logger issues an ENF 48 signal.

If you are not using DSEXTENT in the LOGR, LOGRY or LOGRZ couple data set, the data set directory limit for a log stream is 168 data sets. (Note that your DASD staging data sets **do not** count toward the 168 limit.) If you reach the data set directory limit for a log stream, system logger will not be able to offload data.

You can make additional directory space available for a log stream by deleting log data from the log stream. (See [“Deleting log data and log data sets” on page 291](#).) Whenever the number of log data sets in use gets above 90% of the total number allowed (168), the system logger issues message, IXG257I. This message will be deleted when either the number of log data sets for the log stream drops below 85% of the total allowed or the last connection to the log stream in the sysplex disconnects. In the latter case, the message will be reissued if an application reconnects to the log stream.

Whether or not you use DSEXTENT to increase the number of log data set directory extents for a log stream, you can also set up a retention period and automatic deletion policy for a log stream to ensure that your log streams will not run out of directory space. See [“Managing log data: How much? For how long?” on page 272](#).

Plan space for staging data sets

DASD staging data sets are used for duplexing log data. The way you plan for staging data sets depends on whether you are planning for coupling facility or DASD-only log streams. See:

- [“Plan staging data sets for coupling facility log streams” on page 263](#)
- [“Plan staging data sets for DASD-only log streams” on page 265](#)
- [“Monitoring staging data set usage log streams” on page 266](#)

Plan staging data sets for coupling facility log streams

For coupling facility log streams, staging data sets are optional. However, Logger allows staging data set attributes to be defined and updated for all log streams, whether or not staging duplexing is requested. Therefore, **IBM recommends** that you plan for staging data sets for each system even if your installation does not intend to use them for duplexing and does not define duplexing to staging data sets in the log stream definition. This is recommended because system logger might allocate temporary staging data sets to safeguard log data when there is an error that leaves the only copy of log data in a volatile configuration. Specifically, system logger automatically allocates a staging data set when a structure failure followed by a rebuild failure leaves the only copy of log data for a log stream in local storage buffers. System logger automatically allocates staging data sets for each system not already using them under these conditions, even if you do not specify duplexing to staging data sets. System logger automatically allocates temporary staging data sets during rebuild processing. System logger does the following to allocate staging data sets during rebuild processing:

- Allocates a staging data set on each system.
- Copies the log data into the staging data set.
- Disconnects user connections to the log stream and places the system connection to the log stream into a failed state.

Note that the resulting staging data set size can be up to 16TB - for example, 16384GB or 17,592,186,044,416 bytes. For important information about choosing and testing log data set size changes, see [“Testing log data set parameter modifications” on page 269](#).

For coupling facility log streams, **IBM recommends** that all the staging data sets reside on devices that all systems in the sysplex have connectivity to through channel paths. This is important because other systems might need to access the backed up log data in case of system or coupling facility failure. If peer systems do not have connectivity to staging data sets, system logger might not be able to recover all data in case of a failure.

See [“Selecting a method of duplexing coupling facility log data” on page 283](#) for more planning staging data set duplexing.

Sizing staging data sets for coupling facility log streams

To specify a size for staging data sets for a coupling facility log stream, you can either take the system logger defined default size, or define your own size on the STG_SIZE parameter in the log stream definition in the LOGR policy (for this log stream or the one specified on the LIKE parameter). The staging data set size can also be established via automatic class selection (ACS) routine that SMS uses to assign the constructs for the data set.

Note that whether you take the default staging data size or define your own, the definition applies for all staging data sets for a log stream.

Staging data sets on a system are deleted when the last connector to the log stream for that system disconnects normally.

• Using the system logger default staging data set size:

If you do not define a staging data set size on the STG_SIZE parameter in the LOGR policy, system logger determines the default staging data set for a system from the maximum size of the coupling facility structure allocated for a log stream. This value is defined on the SIZE parameter in the CFRM function couple data set. As a result, a default staging data set is as large as if each system was actually

going to write the amount of data estimated for the entire log stream. If the size of the coupling facility structure is greater than 4GB, you must specify a STG_SIZE value. See "Defining your own staging data set size."

- **Defining your own staging data set size:**

To define your own staging data set size for a log stream, specify it on the STG_SIZE parameter in the log stream definition in the LOGR policy. The value specified on STG_SIZE overrides the system logger default staging data set size. Use the following guidelines to determine the staging data set size for a log stream:

- For log streams that are used by only one system, size the staging data set to be large enough to hold all the log data written by the system logger application. Use the maximum coupling facility structure space allocated for the log stream, specified on the SIZE parameter in the CFRM function couple data set.
- For log streams written to by multiple systems, **IBM still suggests** that you play it safe by sizing the staging data sets for the log stream the same size as the coupling facility space allocated for that log stream on the SIZE parameter in the CFRM function couple data set.

If you need to save DASD consumption by sizing your staging data sets smaller than the recommended size, make sure you at least specify a size large enough for the system writing the most data. For example, if one system writes about 40% of the data to the log stream, while another writes 60%, you must specify a staging data set size for the log stream that will accommodate **at least** 60% of the data. To make this calculation and depend on it, you must know exactly how much data each system writes to the log stream.

Remember also, that if your staging data set is too small, you might find that offloading, with its overhead cost, is kicked off by the high offload threshold in the staging data set, rather than the coupling facility threshold. See [“Staging data set size and offloading” on page 226](#).

IBM recommends that staging data sets be sized no smaller than 10 MB, unless the exploiter specifically recommends a size below this amount. In addition to assisting to prevent frequent offloading, this is intended to assist in avoiding staging data set full conditions. These conditions may lead to poor application performance and outages.

- For either single-system or multiple-system use logstreams, if the log stream is defined to a coupling facility structure greater than 4GB in size, ensure that the staging data set size corresponds to the size of the CF structure.
- If trying to achieve a data set near or greater than 4 GB in size, see [“Testing log data set parameter modifications” on page 269](#) for important notes on choosing and testing data set size changes.

Change the staging data set size for coupling facility log streams

If you need to change the staging data set size for a log stream because of a change in conditions or in response to monitoring SMF record 88, note that the staging data sets must actually be deleted and then reallocated in order to change. This is because the staging data set size and other log stream attributes are defined at allocation and remain in effect as follows:

- For a log stream defined as STG_DUPLEX(YES) and DUPLEXMODE(UNCOND), until the last connection to the log stream disconnects.
- For a log stream defined with STG_DUPLEX(YES) and DUPLEXMODE(COND), while the configuration contains a single point of failure or until the last connection to the log stream disconnects.

To change staging data set size, do the following:

1. Depending on how the log stream is defined, do one of the following:
 - For a log stream defined with STG_DUPLEX(YES) and DUPLEXMODE(UNCOND), have all instances of the application on all systems disconnect from the log stream.
 - For a log stream defined with STG_DUPLEX(YES) and DUPLEXMODE(COND), either issue a structure rebuild so that the configuration no longer contains a single point of failure or else have all instances of the application on all systems disconnect from the log stream.

2. Change the STG_SIZE parameter in the LOGR policy either to a new value or the default, to take the system logger defined default.

Alternatively, change the STG_DATACLAS parameter or if an automatic class selection (ACS) routine is used by SMS, then the ACS routine may need to be updated to assign a new SMS data class construct for the log stream staging data sets.

3. Have all systems reconnect to the log stream.

For example, to change the staging data set size for logrec log streams, you would:

1. Issue the SETLOGRC DATASET command to disconnect logrec from the log stream and record logrec records in a data set.
2. Update the STG_SIZE value in the LOGR policy using the IXGINVNT service or the IXCMIAPU utility.
3. Issue the SETLOGRC LOGSTREAM command to reconnect to and reactivate the logrec log stream.

If the update of the staging data set size does not work, it might be because there are more connectors to the log stream that have not been disconnected. Use the DISPLAY LOGGER,L and DISPLAY LOGGER,C commands with the LSN=*name* specified, or use the LIST request on the IXCMIAPU utility, to see if there are still connectors. Make sure that all log stream connectors have disconnected, by canceling the job if necessary, before changing the staging data set size in the LOGR policy.

Plan staging data sets for DASD-only log streams

For DASD-only log streams, staging data sets are a required part of the system logger configuration. System logger automatically duplexes data to the staging data set for the system at the same time it writes the data to local storage buffers. **IBM suggests** that you plan for staging data sets for each system, using SMS to manage them. If you do not plan for the staging data sets using SMS, system logger will automatically allocate staging data sets for a system connecting to a DASD-only log stream, using the defaults outlined below.

IBM suggests that all the staging data sets reside on devices that all systems in the sysplex have connectivity to through channel paths, if the application intends to have more than one system connect to the DASD-only log stream (in sequence, not simultaneously). If the application intends to have only one system ever connect to a staging data sets, this connectivity is not necessary.

Sizing the staging data sets for DASD-only log streams

To size data sets for a DASD-only log stream, you can either take the system logger defined default size or define your own size on the STG_SIZE parameter in the log stream definition in the LOGR policy. The default minimum size is the recommended minimum of 10MB.

IBM recommends that staging data sets be sized no smaller than 10 MB, unless the exploiter specifically recommends a size below this amount. The **IBM recommended** minimum is intended to assist in avoiding staging data set full conditions. These conditions may lead to poor application performance or possibly exploiter outages.

Note: The staging data set should be sized large enough to allow normal offload activity to keep up with the pace at which IXGWRITE requests are received.

To determine the default staging data set size for a DASD-only log stream, system logger does one of the following, in the order listed:

- Uses the STG_SIZE of the log stream specified on the LIKE parameter, if specified.
- Uses the size defined in the SMS data class for the staging data sets.
- Uses dynamic allocation rules for allocating data sets, if SMS is not available.

Note that if both the STG_DATACLAS and STG_SIZE parameters are specified for a log stream, the value for STG_SIZE overrides the STG_DATACLAS value.

The resulting staging data set size can be up to 16TB - for example, 16384GB or 17,592,186,044,416 bytes. For important notes on choosing and testing log data set size changes, see [“Testing log data set parameter modifications” on page 269](#).

Changing the staging data set size for a DASD-only log stream

If you need to change the staging data set size for a DASD-only log stream because of a change in conditions or in response to monitoring SMF record 88, note that the staging data set must actually be deleted and then reallocated in order to change. This is because the staging data set size and other log stream attributes are defined at allocation and remain in effect until all the connections to the DASD-only log stream have disconnected.

To change staging data set size, do the following:

1. Have all connections to the DASD-only log stream disconnect.
2. Change the STG_SIZE parameter in the active LOGR, LOGRY or LOGRZ policy to a new value.
3. Have all connections from the system reconnect to the log stream.

If the update of the staging data set size does not work, it might be because the system is still connected to the log stream. Use the LIST request on the IXCMIAPU utility to see if the system is still connected. Make sure that the system is disconnected, by canceling the job if necessary, before changing the staging data set size in the LOGR, LOGRY or LOGRZ policy.

Monitoring staging data set usage log streams

Whether your staging data sets are defined by system logger or on the STG_SIZE parameter, you should carefully monitor your staging data sets. This applies to both coupling facility and DASD-only log streams, and is important because the consequences of having your staging data set fill up can be quite disruptive. When a system's staging data set fills up, system logger applications on that system will not be able to write to the log stream until log data can be off-loaded to DASD, which frees up space in the staging data set. Thus, when your staging data sets are too small, system logger will perform coupling facility offloading more frequently than the **HIGHOFFLOAD** and **LOWOFFLOAD** thresholds defined for the log stream would otherwise require. This can negatively affect the performance of all the log streams in that structure.

Use SMF record 88 to monitor staging data set usage:

- Check field SMF88ETT to see whether the high threshold mark for a staging data set is being reached. If you find that the high threshold mark is never reached, your staging data set might be larger than necessary.
- Check field SMF88ETF to see if you are getting one or more staging data set full conditions. This indicates that your staging data set might be too small and should probably be enlarged.

Set up the SMS environment for DASD data sets

System logger requires that you have SMS installed and its address space active at your installation on each system where system logger is expected to run. This is true even if you do not use SMS to manage your volumes and data sets. SMS must be active because system logger uses VSAM linear data sets. If the SMS address space is not active when you attempt to use a system logger application, system logger issues system messages that indicate allocation errors and the application will not be able to use the logger function.

It is recommended that the log stream DASD data sets are SMS-managed.

If you do not use SMS to manage your data sets, you can set up a *minimal* SMS configuration to provide an environment for SMS. The minimal SMS configuration allows you to define model allocation characteristics for data sets. See [z/OS DFSMS Implementing System-Managed Storage](#) for information about establishing a minimal SMS configuration.

Set up the SMS environment to manage DASD log data sets and DASD staging data sets. See [z/OS DFSMSdfp Storage Administration](#) for creating the following SMS classes:

1. Storage groups identifying the volumes to be used for data set allocation.
2. Storage classes to describe the performance characteristics for the data sets.
3. Data classes to specify the data set characteristics and size of the data sets.

4. Management classes to specify the migration characteristics of the data sets. The data sets might be migrated to hierarchical storage manager (HSM) or archived. See [“Log stream data set migration and recall considerations”](#) on page 268.
5. Automatic class selection (ACS) routines to assign the storage groups, classes, data classes and management classes, based on data set name when the DASD data sets are allocated.

You can specify the SMS characteristics for log and staging data sets in a number of ways, depending on how an installation is set up. For example, there might be installation written automatic class selection (ACS) routines to assign classes for DASD data sets, or you can specify the SMS characteristics for DASD log and staging data sets in the LOGR policy. To define the SMS characteristics in the LOGR policy, see [“Specifying SMS data set characteristics for DASD data sets”](#) on page 281.

The EXT (extended format) Data Set Name Type option, in your DFSMS data class that is identified on the log stream LS_DATACLAS or STG_DATACLAS attributes, can be specified to gain the desired data set characteristics. For example, VSAM data striping for either log stream DASD offload, or staging data sets.

Note, however, for log stream offload data sets:

If you specify EXT on the Data Set Name Type in your DFSMS data class that is identified on the log stream LS_DATACLAS attribute, consider specifying Extended Addressability as 'N' to prevent logger from using offload data sets larger than 4GB in size. If you specify Extended Addressability as 'Y', ensure the offload data sets you allocate are less than 4GB.

If the allocation is over 4GB, logger will deallocate the offload data set and try to allocate an offload data set near 3.5GB to maintain log stream availability. See [“Testing log data set parameter modifications”](#) on page 269 for more information.

Log stream staging data sets can be greater than 4GB - for example, up to 16TB (16384GB or 17,592,186,044,416 bytes). Therefore the use of the EXT Data Set Name Type and Extended Addressability as 'Y' combination, can be specified along with the other data set sizing attributes in your DFSMS data class that is identified on the log stream STG_DATACLAS attribute. Also, note that logger requires the Control Interval size (CIsz) for a staging dataset to be 4KB (4096).

Define share options for DASD data sets

If you have multiple systems in the sysplex, it is typical for system logger to require access to log stream data sets and staging data sets from multiple systems. For this reason, you must specify VSAM SHAREOPTIONS(3,3) for log stream data sets and staging data sets.

If you are using a single system sysplex, there is no requirement for any particular SHAREOPTIONS.

- If you use SMS to manage DASD data sets, use the following procedure to define VSAM share options.
To define VSAM share options for a log stream's DASD log data sets and staging data sets, set up a data class and associate it with the DASD log data sets and staging data sets in the LOGR policy. Make this association when you define or update the LOGR policy with the Administrative Data Utility (IXCMIAPU). Use the LS_DATACLAS keyword to make the association. If your sysplex has multiple systems, you must use VSAM SHAREOPTIONS(3,3) so that all the systems in the sysplex can share the data sets.
- If you do not use SMS to manage DASD data sets, your installation must do one of the following:
 - Preallocate two data sets, one with the desired characteristics for log data sets and the other with those for staging data sets. Provide an allocation exit to set the DALLIKE text unit that specifies the correct model data set with the correct SHAREOPTIONS(3,3). See [z/OS MVS Programming: Authorized Assembler Services Guide](#) for more information.
 - Activate a minimal SMS configuration in which the DATACLAS for logger log data sets and staging data sets has been defined. Provide an ACS DATACLAS routine where the DATACLAS specifies SHAREOPTIONS(3,3) for log stream related data sets. See [z/OS DFSMSdfp Storage Administration](#) for more information.

Add the DASD data sets to the GRSRNL inclusion list

Add the DASD log data sets and staging data sets to the GRSRNL inclusion list:

1. Update the GRSRNLxx parmlib member.

Add the following RNLDEF statement in the GRSRNL inclusion list to cover all log stream data sets allocated on behalf of log streams that are defined with the default high level qualifier:

```
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(IXGLOGR)
```

IBM suggests that you add other RNLDEF statements in the GRSRNL inclusion list to cover log streams that are defined with a high level qualifier other than IXGLOGR:

```
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(hlq)
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(hlq.laname)
```

When a single-system scope system logger couple data set type of either LOGRY or LOGRZ is used on a system in the sysplex, IBM recommends that you add the following RNLDEF statements in the GRSRNL exclusion list on that system to cover all log stream data sets allocated on behalf of log streams:

- For log streams that are defined with the default high-level qualifier for a system using the logger single-system scope CDS LOGRY:

```
RNLDEF RNL(EXCL)TYPE(GENERIC)QNAME(SYSDSN)RNAME(IXGLOGRY)
```

- For log streams that are defined with the default high-level qualifier for a system using the logger single-system scope CDS LOGRZ:

```
RNLDEF RNL(EXCL)TYPE(GENERIC)QNAME(SYSDSN)RNAME(IXGLOGRZ)
```

- To cover log streams that are defined with a high level qualifier (HLQ) or extended high level qualifier (EHLQ) when using either logger single-system scope CDS:

```
RNLDEF RNL(EXCL)TYPE(GENERIC)QNAME(SYSDSN)RNAME(hlq.laname)
RNLDEF RNL(EXCL)TYPE(GENERIC)QNAME(SYSDSN)RNAME(ehlq.laname)
```

hlq ehlq

Specifies either a high-level qualifier or an extended high-level associated with a log stream DASD data set or a staging data set. Each log stream DASD data set or staging data set begins with the specified high-level or extended high-level qualifier. The high-level or extended high-level qualifier can be specified when you define a log stream to the LOGR policy.

The high-level qualifier and extended high-level qualifier are mutually exclusive and cannot be specified for the same log stream definition. The extended high-level qualifier requires a LOGR couple data set that is formatted at a z/OS Release 1.2 or higher level.

laname

Specifies the portion of the log stream name that limits the scope of the RNLDEF statement to the system logger-created data sets.

See *z/OS MVS Planning: Global Resource Serialization* for more information.

2. Enter the following command from any system in the sysplex to activate the parmlib updates:

```
SET GRSRNL=(xx,xx,...)
```

Managing logger log stream data sets

The following sections contain additional information about managing logger log stream data sets.

Also see [“Offloading log data from interim storage by freeing and/or moving it to DASD”](#) on page 222 for more information about log stream offload processing.

Log stream data set migration and recall considerations

Logger allows both log stream offload and staging data sets to be migrated. Logger will automatically recall data sets when they are needed. You can migrate data sets using SMS or another protocol.

Some applications may not function properly if long delays are encountered while logger waits for a data set to be recalled. The IXCMIAPU keyword OFFLOADRECALL(NO) specifies that logger is not to recall the current log stream offload data set when performing offload processing for a given log stream. If the current log stream offload data set is migrated, logger will move to a new offload data set by allocating a new one and then continue the offload by writing to the new data set. While this may avoid the delay of recalling the offload data set, it may cause a waste of space on the DASD if the data set is recalled. Therefore care should be taken when using OFFLOADRECALL(YES) to size the data sets accordingly.

With OFFLOADRECALL(YES), logger will not be able to offload until the current offload data set can be recalled. If HSM is not available and either:

- The current offload data set for one or more of the logstreams residing in the structure is migrated while the structure is being rebuilt, or
- Logger is going through recovery for one of the logstreams whose current offload data set is migrated.

The following messages might appear:

- IXG114A OFFLOAD IS NOT PROGRESSING ON sysname LOGSTREAM: logstreamname, STRUCTURE: structurename CHECK FOR OFFLOAD INHIBITORS WITHIN sysname.
- IXG115A CORRECT THE OFFLOAD CONDITION ON sysname FOR structurename OR REPLY TASK=END TO END THE STRUCTURE TASK.

Similarly, if an IXGBRWSE request is made for log stream data that resides in a migrated data set, logger will not be able to process the request until the data set is recalled. Thus, if HSM is down or the recall takes a long time, browse delays may result. The delays are acceptable as long as the application can tolerate them.

IBM strongly suggest that you do not migrate staging data sets and avoiding migrating offload data sets whenever possible. If migration is necessary because of DASD space constraints, migrate the low priority offload data sets for which associated applications can handle delays in accessing the data sets. Use LS_STORCLAS and STG_STORCLAS to specify on which pool or volumes logger offload and staging data sets can be allocated. Use LS_MGMTCLAS and STG_MGMTCLAS to specify if the data sets are migratable or not migratable.

Consider associating the STORCLAS with the MGMTCLAS so that migratable data sets are all allocated in the same pool or on the same volumes.

Copying log stream offload data sets

If you use IDCAMS REPRO (either directly or indirectly through DSS copy) with a log stream offload data set to which the logger function has not fully written, you might receive errors from IDCAMS.

Logger offload data sets are allocated as VSAM linear data sets but are not fully formatted. A tool like IDCAMS REPRO expects data sets to be formatted up to the high used relative byte address (HURBA). Tools such as IDCAMS REPRO that perform logical or sequential data set reads using VSAM access methods fail the copying of log stream offload data sets that are not completely formatted.

If the data set copy fails use a tool such as DSS DUMP/RESTORE that performs a physical copy of the data set. For information on identifying the types of failures that can occur, see [*z/OS MVS Diagnosis: Reference*](#).

Testing log data set parameter modifications

When defining or updating a log stream, it is important to understand what the resulting offload and/or staging data set size will be. Logger provides parameters LS_DATACLAS, LS_MGMTCLAS, LS_STORCLAS, STG_DATACLAS, STG_MGMTCLAS, and STG_STORCLAS to provide SMS data, management, and storage classes for your data sets. Logger also provides the keywords LS_SIZE, and STG_SIZE to override the SMS classes or for use with non SMS managed data sets. The data set attributes may be overridden by DFSMS Automatic Class Selection (ACS) routines, or through dynamic allocation rules. Furthermore, the actual data set size of your logger offload or staging data sets depends on many factors including track size, CI SIZE, and volume type, and may be smaller or larger than your parameter inputs expect.

- Small Data Sets:

The required minimum size that a log stream dataset can be defined for system logger use is 64K bytes. If the data set size is too small, system logger automatically attempts to reallocate it to make it large enough for use. If the size of the data set does not meet the minimum requirement of a valid system logger data set, the system issues message IXG256 to the hardcopy log; this message indicates that system logger is attempting to reallocate the data set within the acceptable bounds. If the reallocation fails, system logger is unable to continue and will issue IXG263 to the hardcopy log.

The 64K required minimum means that any LS_SIZE or STG_SIZE specification should be greater than 16.

If your offload data sets are sized too small, this can drive frequent data set allocations and delay the offload process. In addition, these frequent allocations can cause data set extents to run out, which eventually leads to write failures. Sizing your staging data sets too small can cause the staging data set to be perpetually full. This, in turn, drives constant offload activity and making the log stream frequently unavailable for writes.

If you do not specify data set size parameters in the LOGR policy, allocation will use your system's default, which could be as small as 2 tracks. To avoid problems, IBM recommends that you set the data set size to avoid using the small default value. The recommended minimum size for a staging data set is 10MB, which can be achieved with a STG_SIZE of 2560; the recommended minimum size for a offload data set is 1MB, which can be achieved with an LS_SIZE of 256.

Starting in z/OS Version 2 Release 3, logger system configuration policy options are provided to indicate whether logger ensures log stream offload. It also ensures that staging data sets are newly allocated on the system with the base minimum default sizes. This means that the system ensures that log stream data sets are newly allocated on the system with a base minimum size of at least 1MB and 10MB for offload and staging data sets. For more information about the USEOFFLOADADMIN and USESTAGINGMIN specifications on the respective MANAGE OFFLOAD and MANAGE STAGING statements, see *z/OS MVS Initialization and Tuning Reference* and the IXGCNFxx member in SYS1.PARMLIB.

- Large Data Sets:

Logger supports different data set size maximums for log stream offload and staging data sets. The maximum size for log stream offload data sets is 4GB - 1. If the system logger parameters and other factors cause allocation to attempt to create a log data set greater than or equal to the 4GB size, the request may fail; any attempts to define or connect to the log stream may also fail. If the size change is the result of an update request, system logger may not be able to offload data when it is necessary to allocate a new offload data set.

If the offload data set size is too large, system logger may issue message IXG251I to the hardcopy log containing message IGD306I with return code 140 and reason code 110.

If Extended Addressability is specified on your SMS data class definition, then it is possible for an allocation of a data set greater than 4GB to succeed.

If logger detects that an offload data set is 4GB or greater in size, logger will deallocate the data set and then try to allocate an offload data set near 3.5 GB in size. Logger will ask for 917504 4K segments on the subsequent allocation request.

System logger supports staging data set sizes greater than 4GB. Use the STG_DATACLAS log stream specification and define the corresponding DFSMS data class with space allocation attributes. Also, include the following attributes in the data class definition:

- Data Set Name Type "EXT" (extended format)

Extended format is a means of storing data on a logical DASD volume.

- Extended Addressability "Y"

Extended addressability is a means to allow VSAM data sets greater addressability beyond the 4GB address constraint.

- Performing the test:

Because many factors affect the resulting size of offload and staging log data sets, it is helpful to test changes to the log stream configuration for DASD before you implement the changes on the

production log stream. IBM suggests that you follow this procedure before you make any parameter or configuration changes that can affect the size of the log data sets.

Consider the following actions; these procedures will cause a log stream resource to be in use until you are able to delete the definition.

- **For offload data set configuration changes:** Define a GROUP(TEST) log stream. This log stream should use the same configuration definitions as the production log stream apart from any parameter changes that you are making for testing purposes. For example, unless it is one of the items that you are planning to change, ensure that the LS_SIZE, CI_SIZE, SMS classes, DASD device, and so forth are the same values as those used for the production log stream. If the log stream define request is successful, it is likely that the update to the production log stream will not result in offload allocation failures if the resulting offload data set is allocated with more than the maximum 4GB size.
- **For staging data set configuration changes:** Follow a similar procedure as that for the offload data set; for example, define the same values for STG_SIZE and other parameters for the staging data set as those for production. To test the allocation of a staging data set, you must connect to the log stream. IXGCONLS provides a mechanism for connecting to the log stream. If the log stream connect request is successful, it is likely that the update to the production log stream will also not result in allocation failures of the staging data set. For details about IXGCONLS, see [z/OS MVS Diagnosis: Reference](#).
- **To verify the data set size:** Consider checking one of the following:
 - Message IXG283I in the system log
 - IXCMIAPU DATA TYPE(LOGR) report option LIST LOGSTREAM(lsn) LISTCAT DETAIL(YES)
 - IDCAMS LISTCAT ENTRIES(hlq.lsn.suffix) ALL checking field HI-A-RBA.
- **Approaching the 4G limit:**

If you are attempting to use the LS_SIZE parameter to define a log stream that uses offload data set sizes that are close to the 4GB limit, consider using the values that are shown in [Table 21 on page 271](#) and [Table 22 on page 271](#) as guides.

The observations in [Table 21 on page 271](#) were made on internal IBM test systems running z/OS Version 1 Release 11, using DS8000 DASD, in an attempt to cause Logger to allocate staging and offload data sets close to the 4GB (4,294,967,296 byte) limit.

Table 21. Settings to consider for data sets approaching 4GB - z/OS Version 1 Release 11			
Data Set Type	CI Size	xx_SIZE Parameter	Resulting DS Size
Offload	4K	1048400	4,294,656,000
	24K	872000	4,286,545,920

By comparison, the observations in [Table 22 on page 271](#) were made on internal IBM test systems running z/OS Version 1 Release 13, using DS8300 DASD.

Table 22. Settings to consider for data sets approaching 4GB - z/OS Version 1 Release 13			
Data Set Type	CI Size	xx_SIZE Parameter	Resulting DS Size
Offload	4K	1048400	4,294,656,000
	24K	1048400	4,294,656,000

Because many factors can impact the resulting size of a log stream data set, IBM recommends that you verify that the parameters give you the expected results for your environment. Also, IBM recommends that you do not choose an LS_SIZE value that is intended to get the log stream data sets just below 4GB; in this case, minor configuration changes can result in the attempted allocation of a data set over 4GB.

Managing log data: How much? For how long?

For installations with an active primary LOGR couple data set at an OS/390 Release 3 level or higher, system logger provides support to make it easier to archive log data and manage the amount of data kept in a log stream. This applies to both coupling facility and DASD-only log streams. You can define a retention period and automatic deletion policy for each log stream. The retention period and automatic deletion policy are specified on the RETPD and AUTODELETE parameters in a LOGR couple data set to help manage:

- How much data you keep in a log stream.
- How long you keep data in the log stream.

The RETPD and AUTODELETE parameters are used together to set up a retention period and automatic deletion for a log stream. You can specify the RETPD and AUTODELETE parameters in a LOGR couple data set using either the IXCMIAPU utility or the IXGINVNT service in a program. See [“Add information about log streams and coupling facility structures to the LOGR policy”](#) on page 280.

The RETPD parameter allows you to specify a retention period for a log stream. On RETPD, you specify the number of days that you want to keep data in the log stream, even if the data has been marked for deletion using IXGDELET. For example, if you specify RETPD(7) in the LOGR policy for a log stream, the retention period for data in that log stream is 7 days from the time the data is written to the log stream by the application. System logger processes the retention period on a log data set basis. Once the retention period for the entire log data set has expired, the data set is eligible for deletion. System logger may not physically delete data as soon as it hits the retention period. The point at which system logger physically deletes the data depends on when the data set fills and what you specify on the AUTODELETE parameter.

Note that this retention period is different from a data set retention period on a JCL DD statement; a system logger retention period applies to the age of the log data, not the data set.

The AUTODELETE parameter lets you specify:

- AUTODELETE(NO) to indicate that you do not want an automatic deletion policy for a log stream. AUTODELETE(NO) means that the log data can be physically deleted only after the log data has been marked for deletion via IXGDELET **and** after any retention period specified for the log stream has expired. AUTODELETE(NO) is the default.
- AUTODELETE(YES) to indicate that you want an automatic deletion policy to limit how long data is kept in the log stream. AUTODELETE(YES) means that log data can be physically deleted **either** when the data is marked for deletion or when a retention period specified for the log stream expires. Use care when specifying AUTODELETE(YES). Automatic deletion is designed to speed physical deletion of log data, which can mean deletion of data that an application needs.

Choosing the right retention period and automatic deletion policy for a log stream

The way you use RETPD and AUTODELETE together to set up a retention period and an automatic deletion policy for a log stream depends on the data requirements of each log stream. See [“Deleting log data and log data sets”](#) on page 291.

Example 1. Delete data after no more than 3 days, no matter what

For audit log streams, such as the OPERLOG log stream, the installation must often manage how much data is kept in the log stream. Using automatic deletion and a retention period, you can manage the amount of time log data resides in the log stream, cutting down also on the storage consumed by log data in this log stream. For example, assume that log AUDIT1 requires that data be kept for no more than 3 days and then physically be deleted. For AUDIT1, you would specify RETPD(3) and AUTODELETE(YES) to make sure that data is kept for no more than 3 days.

Example 2. Keep data for at least 30 days, even if it was marked for deletion

Installations may require that their transaction manager log streams retain data for a certain length of time to meet audit requirements. This data must be kept for the retention period, even when it has been

marked for deletion via the IXGDELET service. Using automatic deletion and a retention period, you can archive the data for the required length of time without having to move the data out of the log stream, and without having the archived data interfere with the transaction manager. For example, log TRANS1 needs to have data in the log stream for at least 30 days, even if it has been marked for deletion. You can specify RETPD(30) AUTODELETE(NO) to keep the data for the required amount of time. Even if data is marked for deletion via IXGDELET within the 30 day retention period, it is kept in the log stream and can be accessed by a system logger application using the IXGBRWSE service with VIEW=ALL.

Example 3. Keep data in the log stream until deletion

Some log streams may not need to keep log data around after it has been marked for deletion via IXGDELET and do not require that data be retained for a particular length of time. For this kind of a log stream, the default settings of RETPD(0) and AUTODELETE(NO) should be just fine. System logger will consider data eligible for physical deletion any time after it has been marked for deletion via IXGDELET.

Example 4. Don't keep my log data at all

If you specify RETPD(0) and AUTODELETE(YES), data is eligible for deletion as soon as you write it to the log stream. This setting is dangerous, because data can be physically deleted whether it is marked for deletion or not.



Attention: This setting is dangerous, because data can be physically deleted whether it is marked for deletion or not. If the application using the logstream attempts to read data after the RETPD / AUTODELETE setting has already deleted the data, an ABEND1C5 RSN00000804 reflecting the "missing" data will be issued.

Using a resource manager and auto deletion together

If you have specified AUTODELETE(YES) for a log stream, and you also manage that log stream with a resource manager and a resource manager exit, note that automatic deletion processing takes precedence over the delete override processing performed by the resource manager exit. This means that log data can be deleted even when a delete request is rejected or overridden by the resource manager user exit. In general, IBM suggests AUTODELETE(NO), especially for log streams that are backed up by a resource manager.

Define authorization to system logger resources

IBM suggests that installations use System Authorization Facility (SAF) to control access to system logger resources, such as log streams or coupling facility structures associated with log streams. The following examples assume the use of z/OS Security Server (RACF) or an equivalent security product.

Installations must define authorization to system logger resources for the following:

- System logger address space
- User ID setting up the CFRM and LOGR policies
- System logger applications (for example, logrec or OPERLOG) that will be accessing the log streams.

These steps apply to both coupling facility and DASD-only log streams.

For information about the authorization an application needs to issue system logger services, see [z/OS MVS Programming: Assembler Services Guide](#).

Define authorization for the system logger address space

IBM suggests that the system logger address space (IXGLOGR) be assigned privileged and/or trusted RACF status. After assigning the system logger address space with the privileged and/or trusted status, you must restart the system logger address space for the newly assigned authorization to take effect. You can stop and restart the system logger address space with:

```
FORCE IXGLOGR,ARM
S IXGLOGRS
```

If the system logger address space for your installation is neither privileged nor trusted, make sure to give IXGLOGR the following SAF authorizations:

- Define IXGLOGR in either RACF started procedures table (SPT) or in RACF CLASS(STARTED).
- For coupling facility log streams, define alter access to RESOURCE(*IXLSTR.structure_name*) CLASS(FACILITY) for access to the log stream coupling facility structures.
- Define alter access to RESOURCE(*hlq.data_set_name*) CLASS (DATASET) for each DASD log stream and staging data set.

Note that the RESOURCE data set could also be identified by an extended high-level qualifier, such as RESOURCE(*ehlq.data_set_name*).

- When using data set level encryption for log stream data sets:
 - Define read access to RESOURCE(CSFKRR2) CLASS(CSFSERV).
 - Define read access to the profiles that cover any ICSF key labels used to encrypt log stream data sets in the general CSFKEYS class.

Refer to [“Considerations for encrypting log stream data sets”](#) on page 236.

- Define read access to RESOURCE(*sys1.parmlib_data_set_name*) CLASS (DATASET) for access to SYS1.PARMLIB.
- Define read access to RESOURCE(MVS.DISPLAY.LOGGER) CLASS (OPERCMDS) when an IXGCNFxx parmlib member is used during IPL or by a SET IXGCNF command, or when parmlib updates are specified on a SETLOGR command.
- Define update access to RESOURCE(MVS.TRACE.CT) CLASS (OPERCMDS) when an IXGCNFxx parmlib member with a CTRACE(*parmlib_member_name*) specification is used during IPL or by a SET IXGCNF command, or when the SETLOGR CTRACE command is used.

Enabling z/OS as a z/OS IBM zAware log stream client requires that the IXGLOGR address space has security permission for an OMVS segment. The OMVS segment is only for TCP/IP connectivity. This requires that either UID(0) be used or superuser authority be granted to the ID.

For example, in RACF issue the following command, where xxxx is a unique user ID:

```
ADDUSER IXGLOGR OMVS(UID(XXXX) HOME('/'))
```

See [“Preparing for z/OS IBM zAware log stream client usage”](#) on page 311 for additional details.

Define authorization for setting up policies

The LOGR policy information describes the characteristics of each log stream and coupling facility structure that will be used when there are active connections to the log stream.

System logger also supports up to two single-system scope policies in a sysplex in addition to the sysplex scope LOGR policy. The LOGRY and LOGRZ policy are each one of the two system logger policies that are single-system scope. You can define the policy information that describes the characteristics of a log stream that will be used when there are active connections to the log stream from a single-system within the sysplex. These log streams are separate resources from the log stream resources defined in the sysplex scope LOGR policy and the other single-system scope policy.

The CFRM policy contains the definition of all coupling facility structures used by system logger, if applicable.

Before setting up these policies for your installation with the IXCMIAPU Administrative Data Utility program, authorize who can use IXCMIAPU.

- Define a resource profile for the resource name MVSADMIN.LOGR to the FACILITY class. Assign READ access authority to users who require reports on the policy, but who will not change the policy.
- If applicable, define a resource profile for the resource name MVSADMIN.LOGRY to the FACILITY class. Assign READ access authority to users who require reports on the policy, but who will not change the policy.

- If applicable, define a resource profile for the resource name MVSADMIN.LOGRZ to the FACILITY class. Assign READ access authority to users who require reports on the policy, but who will not change the policy.
- If applicable, define a resource profile for the resource name MVSADMIN.XCF.CFRM to the FACILITY class. Assign UPDATE access to users who must define coupling facility structures in the policy.

The users of the IXCMIAPU utility require authorization to the resources accessed by the utility. Define the appropriate authorizations before users attempt to add, update, delete, or extract a report from the system logger LOGR, LOGRY or LOGRZ policy. The appropriate system logger couple data set must be activated and available on the respective system(s) before you add log streams and structure definitions to it. The following authorizations are required to set up the LOGR policy:

- To DEFINE, UPDATE, or DELETE LOGSTREAM, specify ALTER access authority to RESOURCE(*log_stream_name*) CLASS(LOGSTRM).
- To DEFINE or DELETE STRUCTURE, specify ALTER access authority to RESOURCE(MVSADMIN.LOGR) CLASS(FACILITY).
- To specify a structure name on a LOGSTREAM definition or update request, (for example,

```
DEFINE LOGSTREAM NAME (log_stream_name) STRUCTNAME(structname)
```

), specify UPDATE access authority to RESOURCE(IXLSTR.*structurename*) CLASS(FACILITY).

- To specify ZAI(YES) on a LOGSTREAM definition or update request (for example,

```
DEFINE LOGSTREAM NAME (log_stream_name) ZAI(YES)
```

specify UPDATE access authority to RESOURCE(IXGZAWARE_CLIENT)CLASS(FACILITY).

Authorization for system logger applications

For an installation or vendor-written system logger application, the access to system logger resources, such as log streams or coupling facility structures associated with log streams, depends on which system logger services the application issues.

- For a vendor-written application, see the documentation provided by the vendor.
- For an installation-written application, see the application programmer for authorization requirements of the program.

Information about system logger services is provided in [*z/OS MVS Programming: Assembler Services Guide*](#).

For information on the authorization requirements for a program connecting to a log stream and the log stream access services that can be used for the different types of connections see 'Requesting authorization to the log stream for an application' in [*z/OS MVS Programming: Assembler Services Guide*](#).

Refer to each log stream exploiter's planning and/or guidance documentation on their log stream access requirements. Grant permissions to SAF profile covering the (WRITE_ONLY_*log-stream-name*) resource only when deemed appropriate for the log stream exploiter.

For logrec log stream, CICS log manager, and OPERLOG system logger application, you must do the following:

- Define UPDATE access to SAF resource RESOURCE(*log_stream_name*) CLASS(LOGSTRM).

Log stream users can specify a log stream subsystem exit routine name to receive control for reading log data through either of the following methods:

- As the SUBSYS=(LOGR,exit_routine_name,...) keyword of the DDNAME JCL statement, or
- On a dynamic allocation request using text units DALSSNM (value LOGR) and DALSSPRM (value specifying the exit_routine_name).

If your installation does not intend to use any log stream subsystem exit routines, or if your installation intends to use only the log stream subsystem exit routine names IXGSEXIT, IFASEXIT, IFBSEXIT, or DFHLGCNV, there is no need to define the RACF policies described here.

If your installation uses or intends to use log stream subsystem exit routine names, other than the set of explicit names listed above, you or your security administrator needs to define a RACF authorization profile in the FACILITY class to cover the resource IXGLOGR.SUBSYS.LSEXIT.exit_routine_name, where exit_routine_name identifies the name of the log stream subsystem exit routine. For more information on defining RACF profiles, see [Planning for profiles in the FACILITY class in z/OS Security Server RACF Security Administrator's Guide](#).

IBM recommends that you do one of the following:

1. Define a discrete profile IXGLOGR.SUBSYS.LSEXIT.exit_routine_name for the FACILITY class to cover the resource, where exit_routine_name identifies the name of the log stream subsystem exit routine. This profile should audit all failures and allow all users READ access.

For example:

```
RDEFINE FACILITY IXGLOGR.SUBSYS.LSEXIT.exit_routine_name
```

```
UACC(READ) AUDIT(FAILURES(READ))
```

2. If you choose to allow for exit_routine_names that may not be explicitly known to be used on your system, meaning that you did not define explicit discrete profile(s) as described in step 1, consider also defining a generic profile IXGLOGR.SUBSYS.LSEXIT.* for the FACILITY class to cover the resources associated with using these log stream subsystem exit routines. Include in this generic profile the WARNING attribute.

For example:

```
RDEFINE FACILITY IXGLOGR.SUBSYS.LSEXIT.*
```

```
UACC(NONE) WARNING
```

When this generic profile is used to cover the authorization check for a resource IXGLOGR.SUBSYS.LSEXIT.exit_routine_name, if the check fails since WARNING has been specified, RACF issues the appropriate warning message to the user, logs the access attempt, and allows the user to access the resource.

This generic profile approach is recommended only as a temporary mechanism to gather information on the possible exit routine names that need to be supported, and once known, you can then define the appropriate discrete profiles. Once the known exit routine names are covered by discrete profiles, delete the IXGLOGR.SUBSYS.LSEXIT.* generic profile.

Note: If you do not define any of the profiles similar to those stated but you have defined a different generic profile that will cover the resource IXGLOGR.SUBSYS.LSEXIT.exit_routine_name, those generic profile attributes will determine the outcome of the authorization checking, logging, and whether the exit_routine_name will be used.

Format the sysplex scope LOGR couple data set and make it available to the sysplex

Do the following for both coupling facility and DASD-only log streams to format a primary and alternate sysplex scope LOGR couple data set and define them to the sysplex:

1. **Format the primary and alternate LOGR couple data set using the IXCL1DSU utility.**

Format a primary and alternate LOGR couple data set using the IXCL1DSU utility. IBM recommends they are on different volumes.

You must provide as input the maximum number of log streams and structure definitions you plan to define. For DASD-only log streams, you will not have structures to define, so do not increase the

number of structures for DASD-only log streams. These values were calculated in [“Plan the system logger configuration”](#) on page 247.

IBM recommends using the highest format level of the LOGR couple data set (CDS) that can be used by the lowest system release level in the sysplex. Currently, the highest LOGR CDS format level is HBB7705. This will allow for the latest logger features to be available given the sysplex configuration. See [“LOGRY or LOGRZ couple data set versioning - new format levels”](#) on page 327 to understand the supported system logger functional capabilities for the different LOGR couple data set format levels.

For complete information and examples of using the IXCL1DSU utility, see [Chapter 11, “Format utility for couple data sets,”](#) on page 313.

2. Update the COUPLExx member in SYS1.PARMLIB to identify the LOGR couple data set to the sysplex.

Add your primary and alternate LOGR couple data set names to the COUPLExx parmlib member. See the following for more information:

- [“Defining function couple data sets to MVS”](#) on page 37 for an example of defining the LOGR couple data set in the COUPLExx parmlib member.
- [z/OS MVS Initialization and Tuning Reference](#) for complete the COUPLExx parmlib member.

3. Make the LOGR couple data set available.

You can use either of the following ways to make the primary and alternate couple data sets available to the sysplex:

- IPL the system with the primary and alternate LOGR couple data sets defined in the COUPLExx parmlib member.
- Issue the following SETXCF commands when you want to bring the LOGR couple data sets online without IPLing the system as follows:

```
SETXCF COUPLE,TYPE=LOGR,PCOUPLE=(primary_couple_data_set)
SETXCF COUPLE,TYPE=LOGR,ACOUPLE=(alternate_couple_data_set)
```

Either of these methods causes the empty LOGR couple data set to become available on all systems where system logger is installed. After the SETXCF command is entered, you can define structures and log streams in the LOGR policy.

4. Additional Guidance on Handling LOGR Couple Data Sets

See [“LOGR couple data set use considerations”](#) on page 325 for guidance on identifying and using an existing LOGR couple data set for use by the sysplex.

Prevent a z/OS image from accessing sysplex scope LOGR couple data sets

System logger requires the use of the sysplex scope LOGR couple data set (CDS) to perform its operations. However, certain sysplex configurations can benefit from a z/OS system that does not maintain any access to these sysplex scope primary or alternative LOGR CDSs. An example of such a system is the K-systems controlling systems in an IBM GDPS/PPRC, GDPS/MTMM environment. For these situations, logger provides for a SYS1.PARMLIB option that you can specify during a restart of a z/OS image. The option directs system logger and cross-system coupling facility (XCF) to not maintain any access to the LOGR CDSs on that z/OS image. For specific details, see the IXGCNFxx parmlib member MANAGE statement and LOGRCDS ALLOWACCESS options in [z/OS MVS Initialization and Tuning Reference](#).

When ALLOWACCESS(NO) is specified, logger indicates to XCF during the processing of the system restart to not maintain access to the sysplex scope LOGR CDSs. Logger message IXG078I and XCF message IXC255I are revealed during the system restart, indicating that the LOGR CDSs are not used on that system. Subsequent SETXCF COUPLE,TYPE=LOGR commands result in XCF message IXC255I indicating that the LOGR CDSs are still not in use on that system. Additionally, logger and XCF messages that are issued in response to respective display commands D LOGGER,ST and D XCF,C,TYPE=LOGR also reveal that the LOGR CDSs are not used on that system.

It is possible to dynamically change ALLOWACCESS(NO) to ALLOWACCESS(YES) after the system restarts, if the system logger address space remains active. See the SET IXGCNF=xx and SETLOGR MANAGE commands in *z/OS MVS System Commands* for more information.

If the ALLOWACCESS option is dynamically changed to YES, then, when subsequent SETXCF COUPLE,TYPE=LOGR commands are run, XCF and logger attempt to use the LOGR CDS configuration that is already in use by the rest of the sysplex, regardless of what is specified on the SETXCF COUPLE command.

However, if the IPL option of ALLOWACCESS(NO) is used with either **USECDSTYPE(LOGRY)** or **USECDSTYPE(LOGRZ)**, meaning with a single-system scope CDS data type, then the ALLOWACCESS(NO) specification cannot be changed to **YES** on SET IXGCNF and SETLOGR MANAGE commands or if the IXGLOGR address space is later restarted. If the sysplex scope **LOGR** CDS data type is intended to be used, then a re-IPL of the system is needed with the options ALLOWACCESS(YES) USECDSTYPE(LOGR) specified. For additional details on using LOGRY and LOGRZ couple data set types, see [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex”](#) on page 279.

Notes:

- The data set name on the command is used only if the entire sysplex is not currently using any LOGR CDSs.
- If the IXGLOGR address space is restarted after the ALLOWACCESS option was dynamically changed to YES, then the setting will remain persistent as YES.
- Logger does not support dynamically changing the ALLOWACCESS option from YES to NO after the system restart.

If the ALLOWACCESS(NO) specification is used for a restart of a z/OS image, then, when that system is restarted again, access to the LOGR CDSs primarily depends upon the combination of the following factors:

- whether there are any active systems in the sysplex,
- the SYS1.PARMLIB specifications during this most recent system restart, and
- whether the restarted z/OS image has physical access to the LOGR CDSs that are requested during this system restart.

The ALLOWACCESS value that is set during subsequent system restarts dictates LOGR CDS access:

ALLOWACCESS(NO)

When ALLOWACCESS(NO) is still specified during the system restart, then access to the LOGR CDSs is not maintained for that z/OS image.

ALLOWACCESS(YES)

When the specification is changed to ALLOWACCESS(YES) for the system restart, then access to the LOGR CDSs can occur. The LOGR CDSs that are used depends upon the previously noted factors.

When the sysplex contains active systems other than the restarting system, then the primary and alternative LOGR CDSs that are currently in use for the sysplex are also used on the restarting system.

If the sysplex contains no other active systems when the restarting system restarts, meaning that it is the first system that is active again in the sysplex, then the primary and alternative LOGR CDSs that are used are determined according to the following criteria:

- When there is a DATA TYPE(LOGR) PCOUPLE(...) specification in the COUPLExx member, then XCF requests that the installation's operations designate which LOGR CDSs are used in the sysplex. Therefore, either the current LOGR CDSs that were in use for the sysplex before this restarting system was restarted or the LOGR CDSs that were specified in the COUPLExx member are used. See messages IXC287I, IXC288I, and IXC289D for other details.
- When there is no DATA TYPE(LOGR) PCOUPLE(...) specification, or if the specification is for the current primary and alternative LOGR CDSs that are in use for the sysplex and there is physical access to them, then the restarting system continues to use the primary and alternative LOGR CDSs that are currently in use for the sysplex.

- If there was no access from any systems in the sysplex to any LOGR CDSs, then the LOGR CDSs that are specified in the COUPLExx member for the restarting system are used in the sysplex.

Note: During a recovery failover event, refer to XCF messages IXC404I and IXC405D and the response "I" to initialize the sysplex.

Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex

When certain z/OS images in a sysplex are configured to not access the system logger sysplex scope LOGR couple data sets, those z/OS images can make use of system logger functions by accessing a single-system scope couple data set type. Each of the CDS data types LOGRY and LOGRZ are allowed to be defined and used by one z/OS image (system) within the sysplex. An example of such a system is the K-systems controlling systems in an IBM GDPS/PPRC, GDPS/MTMM environment. For these situations, logger provides for SYS1.PARMLIB options that you can specify during an IPL of a z/OS image. The options direct system logger and cross-system coupling facility (XCF) to not maintain any access to the sysplex scope LOGR couple data sets on that z/OS image. The options also direct system logger to allow access and use of either (but not both) the LOGRY or LOGRZ couple data sets.

System logger services will be available on that single-system with an isolated system view of the log stream resources within the sysplex. The system logger activity associated with the log streams defined in the LOGRY or LOGRZ couple data sets are managed separately from any log streams being used by other systems in the sysplex. For specific details, see the IXGCNFxx parmlib member MANAGE statement and LOGRCDS ALLOWACCESS and USECDSTYPE options in *z/OS MVS Initialization and Tuning Reference*.

Do the following for both coupling facility and DASD-only log streams to format a primary and alternate system logger single-system scope for either the LOGRY or LOGRZ couple data set type and define them to the sysplex, you must:

1. Format the primary and alternate LOGRY or LOGRZ couple data set using the IXCL1DSU utility.

Format a primary and alternate LOGRY or LOGRZ couple data set using the IXCL1DSU utility. IBM recommends that they are on different volumes.

You must provide the maximum number of log streams definitions that you plan to define. There will be no structures to define because just DASD-only log streams are allowed for these single-system scope CDS data types. These values were calculated in [“Plan the system logger configuration”](#) on page 247.

IBM recommends using the highest format level of the LOGRY and LOGRZ couple data set (CDS) that can be used by the lowest system release level in the sysplex. The current highest LOGRY and LOGRZ CDS format level is HBB77C0 (the Base format level).

For complete information and examples of using the IXCL1DSU utility, see [Chapter 11, “Format utility for couple data sets,”](#) on page 313.

2. Update the COUPLExx member in SYS1.PARMLIB to identify the LOGRY or LOGRZ couple data sets to be used on the z/OS image within the sysplex.

Add your primary and alternate LOGRY or LOGRZ couple data set names to the COUPLExx parmlib member. See the following for more information:

- For an example of defining the LOGRY or LOGRZ couple data set in the COUPLExx parmlib member, see [“Defining function couple data sets to MVS”](#) on page 37.
- For complete description of the COUPLExx parmlib member, see *z/OS MVS Initialization and Tuning Reference*.

3. Make the LOGRY or LOGRZ couple data set available.

You can use either of the following ways to make the primary and alternate couple data set available to the sysplex:

- IPL the system with the primary and alternate LOGRY or LOGRZ couple data sets defined in the COUPLExx parmlib member.

- Issue the following SETXCF commands when you want to bring the LOGRY couple data sets online without IPLing the system as follows:

```
SETXCF COUPLE,TYPE=LOGRY,PCOUPLE=(primary_couple_data_set)
SETXCF COUPLE,TYPE=LOGRY,ACOUPLE=(alternate_couple_data_set)
or
SETXCF COUPLE,TYPE=LOGRZ,PCOUPLE=(primary_couple_data_set)
SETXCF COUPLE,TYPE=LOGRZ,ACOUPLE=(alternate_couple_data_set)
```

Either of these methods causes the empty LOGRY or LOGRZ couple data set to become available only on the system (z/OS image) where system logger is installed and the IXGCNFxx specification indicates USECDSTYPE for the couple data set type. After the SETXCF command is entered, you can define the DASD-only log streams in the LOGRY or LOGRZ policy.

4. Define system logger policy information in LOGRY or LOGRZ keywords.

For details on specifying system logger policy information in a single-system scope couple data set, see [“LOGR keywords and parameters for the administrative data utility”](#) on page 358.

5. See additional guidance on handling LOGRY and LOGRZ Couple Data Sets

- Existing utilities can be used for log stream data access for log streams defined in one of the new system logger couple data set types:
 - these utilities need to be run on the same system that is using LOGRY or LOGRZ CDS data type
 - the utility should move the log data into archive/offload data set(s)
 - the archive/offload data set(s) must be accessible to the other systems in the sysplex in order to merge this log data with the sysplex view
- As with the existing system logger requirement for a consistent configuration using system logger LOGR CDS, when a system logger CDS data type of either LOGRZ or LOGRY is to be used on a z/OS image other than on one that had previously used it, all the logger resource configuration data must be in a time consistent state. Otherwise, system logger will not be able to maintain an accurate view of existing log stream data on the new system.
- On any system in the sysplex:
 - Existing utilities can be used on any system in sysplex that has access to archive/offload data sets in order to merge this log data with the sysplex view.
 - A sort program with the archive/offload data sets can be used to merge data from the system(s) that used the new system logger CDSs (LOGRZ and LOGRY).

For guidance on identifying and using an existing LOGR couple data set for use by the sysplex, see [“LOGR couple data set use considerations”](#) on page 325.

Add information about log streams and coupling facility structures to the LOGR policy

Define each log stream and coupling facility structure in the LOGR policy using the IXCMIAPU utility. Note that each coupling facility structure that you define in the LOGR policy must also be defined in the current CFRM policy for the sysplex. Structure names in the LOGR policy **must match** those defined in the CFRM policy.

When using a system logger single-system scope LOGRY or LOGRZ policy, define each log stream using the IXCMIAPU utility for the respective CDS data type.

See [Chapter 12, “Administrative data utility,”](#) on page 335 for using the IXCMIAPU utility. Use the guidance included throughout this planning topic to help you determine what parameters to specify in the LOGR policy. Guidance for SMS parameters and selecting a method of duplexing log data for your configuration are covered below.

Specifying whether the log stream is DASD-only

This step applies to both the DASD-only and coupling facility log streams.

For each log stream, you must decide whether to define it as DASD-only or coupling facility (see [“Determine the right kind of log stream for each application”](#) on page 247). Use the DASDONLY parameter on a DEFINE LOGSTREAM request as follows:

To define a log stream as a coupling facility log stream, specify or default to DASDONLY(NO) on the log stream definition. You must also specify a structure name on the STRUCTNAME parameter in the definition for a coupling facility log stream.

To define a log stream as DASD-only, specify DASDONLY(YES) on the definition for a log stream that is not associated with a coupling facility structure. When you define a DASD-only log stream, you can also specify MAXBUFSIZE on the DEFINE LOGSTREAM request to define the maximum buffer size that can be written to the DASD-only log stream.

Specifying the average log block size

This step applies only to coupling facility log streams, DASD-only log streams may omit this step.

For each coupling facility structure associated with a coupling facility log stream or streams, use the AVGBUFSIZE parameter to specify the average log block size that system logger applications will write to log streams associated with the structure. System logger uses AVGBUFSIZE to calculate the initial entry-to-element ratio for a coupling facility structure for a log stream.

System logger only uses the AVGBUFSIZE specified initially to set the entry-to-element ratio for the structure. After that, system logger will automatically manage the entry-to-element ratio for a structure dynamically, based on actual structure usage.

To find the current entry-to-element ratio system logger is using for a structure associated with a log stream, use the REPORT or LIST parameters in the IXCMIAPU utility. See [Chapter 12, “Administrative data utility,”](#) on page 335 for information.

If you would like to monitor the changes to the entry-to-element ratio system logger is maintaining, use the aggregate structure information in SMF record type 88. Using this record, you can:

- Monitor the number of entry-to-element ratio changes.
- See the time of the last entry-to-element change.
- See the allocated entry-to-element ratio before the last change.
- See the new entry-to-element target ratio after the last change.

See [z/OS MVS System Management Facilities \(SMF\)](#) for details about SMF record type 88.

For a logrec log stream, **IBM suggests** an AVGBUFSIZE of 4068.

For an OPERLOG log stream, **IBM suggests** an AVGBUFSIZE of 508.

If you select a value for AVGBUFSIZE that is different from the actual average block size, structure space will be wasted. If you run out of structure entries for a structure, a structure full condition will result, even if you have many unused elements in the structure. This disruptive condition will cause frequent offload processing for the structure.

IBM suggests that you do not try to manage the entry-to-element ratio for system logger coupling facility structures using the IXLALTER service. System logger will manage the ratio for optimal structure usage; using IXLALTER will just conflict with system logger's entry-to-element ratio.

For more the directory-to-entry ratio see the connection services topic in [z/OS MVS Programming: Sysplex Services Guide](#).

Specifying SMS data set characteristics for DASD data sets

When you define or update a log stream definition in the LOGR policy, you can assign the SMS storage class, data class, and management class for both the DASD log data sets and staging data sets. For

example, you can specify parameters STG_DATACLAS and LS_DATACLAS in the LOGR policy to assign a data class with the correct VSAM share options. These parameters intersect with other parameters and the system environment in ways that might produce unpredictable results if you do not plan well. So, think carefully before specifying the SMS parameters in the LOGR policy. For instance:

- **SMS parameters and ACS routines:**

If you use installation-written automatic class selection (ACS) routines to assign classes for these DASD data sets, the ACS class might override SMS classes you explicitly specify in the LOGR policy using the IXCMIAPU utility. Make sure you coordinate your ACS routines and the SMS characteristics you specify in the LOGR policy to avoid surprises.

- **Using SMS parameters and the LIKE parameter together:**

If you are using the LIKE parameter for a log stream, note that any explicit definitions for the log stream you are defining override the definitions of the log stream referenced on the LIKE parameter. System logger **will not use** an SMS class from a log stream referenced on a LIKE parameter if there is any value specified for that class on an SMS parameter.

- **Optimizing Control Interval Size:**

Increasing the size of the control interval for DASD log data sets can provide performance improvements in the areas of record processing speed and storage requirements. You can modify the control interval size attribute of DFSMS Data Classes defined for DASD log data sets to gain performance improvements in I/O operations associated with the offloading of log stream data as well as during browse processing when retrieving log stream data from DASD log data sets.

A VSAM linear data set requires a control interval size of from 4096 to 32768 bytes, and can be expressed in increments of 4096 bytes. The VSAM default is 4096 bytes. System logger is designed to obtain optimal I/O performance when DASD log data sets reside on devices with 3390 formats. IBM suggests a control interval size of 24576 bytes to realize optimal I/O performance if system logger DASD log data sets reside on devices with 3390 formats. Specify a DFSMS data class that is defined with a control interval size of 24576 on the LS_DATACLAS parameter of a log stream definition to have DASD log data sets allocated with control interval sizes of 24576. Using a control interval size other than 4096 in conjunction with the LS_SIZE parameter can result in a slightly different DASD space allocation than expected. Confirm that the actual data set size is suitable for your log stream.

Note: Do not change the Control Interval Size attributes for staging data sets to be anything other than 4096. System logger requires that the Control Interval Size for staging data sets be set to 4096. If staging data sets are defined with characteristics that result in a Control Interval Size other than 4096, system logger will not use staging data sets to keep a duplicate copy of log data. Operations involving staging data sets defined with a Control Interval Size other than 4096 will fail to complete successfully.

If you intend to use control interval sizes other than 4096, you might need to modify the STG_DATACLAS specifications or change your DFSMS ACS routine. This action may be required regardless of whether or not you specify STG_DUPLEX (YES) for your log streams. If you explicitly provide a STG_DATACLAS specification for the log stream, ensure the DFSMS data class uses a control interval size of 4096. If you use an ACS routine to have attributes assigned to any log stream data sets, you must ensure the staging data sets do not use a data class that has a control interval size other than 4096. For example, you might need to change the ACS routine to filter based on the data set name up to and including the last qualifier in order to recognize staging data sets.

If your IXCMIAPU request does not complete successfully, you might get return and reason codes returned in the output. These return and reason codes are documented along with IXGINVNT in [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).

Before using the IXCMIAPU utility, you will need to define the SAF authorizations for system logger applications. See [“Authorization for system logger applications” on page 275](#).

Selecting a method of duplexing coupling facility log data

This step applies to coupling facility log streams only. For DASD-only log streams, duplexing to staging data sets is automatic.

Using the STG_DUPLEX and DUPLEXMODE parameters in the LOGR policy, you can request how you want system logger to duplex coupling facility resident log data for a log stream. You can duplex coupling facility data in either local storage buffers for each system or in DASD staging data sets. With z/OS V1R2, another available option is to use the system-managed duplexing rebuild function for duplexing the coupling facility resident log data. The Logger log stream LOGGERDUPLEX parameter and your parallel sysplex configuration will determine whether Logger will duplex the log data or whether it uses the system-managed duplexing rebuild process using duplex-mode coupling facility structures.

The method you choose depends on whether your use of the log stream can tolerate a loss of data, whether your connection contains a single point of failure, and for duplex-mode structures whether there is a failure-independent relationship between two structure instances.

What is a single point of failure?

A **single point of failure** is an environment where one failure can result in the simultaneous loss of both the coupling facility list structure for a log stream and the local storage buffer copy of the data on the system making the connection. If you were using local storage buffers to duplex data, this can result in loss of log data. When your configuration contains a single point of failure, you can safeguard your data by duplexing data to staging data sets.

A single point of failure can also exist for a duplex-mode structure where one failure can result in the simultaneous loss of both structure instances. The two structure instances in this environment are failure-dependent, meaning a single failure will cause the structure data to be lost, even though it is in two copies of the structure. Since a single point of failure can cause the loss of all the log data in the failure-dependent duplex-mode structure, Logger will continue to safeguard the data by using one of its own duplexing methods.

When a configuration is not vulnerable to a single point of failure, it is **failure independent**.

System logger evaluates a connection for a single point of failure based on the location of the coupling facility and its status of volatile or non-volatile. The coupling facility can reside in one of the following configurations:

- The coupling facility executes in a logical partition or LPAR (virtual server), sharing a central processing complex (CPC) with a system that is connected to a log stream associated with the coupling facility.
- The coupling facility is in a stand-alone CPC and does not share the CPC with a system with a connection to the log stream.
- The coupling facility executes as an Integrated Coupling Migration Facility (ICMF) partition on a CPC. Only connections from a system on the same CPC can communicate with this coupling facility.

System logger uses the following rules for determining whether a log stream connection contains a single point of failure:

- **When a coupling facility shares a CPC** with a system that connects to log streams mapping to the same coupling facility, each active connection between the system and the coupling facility contains a single point of failure, regardless of the volatility state of the coupling facility. The connections have a single point of failure because if the CPC should fail, coupling facility resident log data will be lost when both the system, with local storage buffers, and the coupling facility fail with the CPC.
- **When a coupling facility (or the composite view of a duplex-mode structure) is separate** from the system connected to the associated log stream, the volatility status of the coupling facility determines whether the connection is vulnerable to a single point of failure:
 - If the coupling facility (or the composite view of a duplex-mode structure) is non-volatile, the connection is failure independent.

- If the coupling facility (or the composite view of a duplex-mode structure) is volatile, the connection is vulnerable to a single point of failure. A power failure could knock out both the system, with its local storage buffer copy of coupling facility data, and the coupling facility.

A connection is failure independent when it is from a system on a separate CPC from the non-volatile coupling facility to which it is connected.

If a connection contains a single point of failure and STG_DUPLEX(YES) DUPLEXMODE(COND) has been specified in the log stream definition, system logger will duplex coupling facility log data to staging data sets for that connection. If a coupling facility or CPC failure occurs, system logger can retrieve lost coupling facility log data from the staging data sets for a connection with a single point of failure.

Using the STG_DUPLEX and DUPLEXMODE parameters

The STG_DUPLEX and DUPLEXMODE parameters are as follows:

- STG_DUPLEX(NO), which is the default, specifies that system logger is to maintain a copy of coupling facility data in local storage buffers on each system.

Choose this option when your application is not vulnerable to data loss that might occur if your environment contains a single point of failure.

Note that in certain rebuild error cases, system logger will automatically allocate a staging data set, even when you specify STG_DUPLEX(NO). See [“Plan space for staging data sets” on page 263](#) for more information.

- STG_DUPLEX(YES) specifies that you want system logger to maintain a copy of coupling facility data in DASD staging data sets, one per log stream per system. See [“Plan staging data sets for coupling facility log streams” on page 263](#) for more planning for staging data set duplexing.

You can only specify DUPLEXMODE with STG_DUPLEX(YES). Choose one of the following:

- DUPLEXMODE(COND), which is the default, specifies that system logger is to conditionally duplex coupling facility log data for a system **ONLY** when a system's connection to the log stream contains a single point of failure and is therefore vulnerable to loss of data.

With DUPLEXMODE(COND), note that different systems connected to a log stream might be backing up data differently. Duplexing is done on a connection basis, depending on whether a system's connection contains a single point of failure.

If you have specified STG_DUPLEX(YES) DUPLEXMODE(COND), system logger can switch duplexing methods when the system logger configuration changes. For example, if you specify conditional duplexing for a log stream that writes to a coupling facility that changes from volatile to non-volatile, system logger will stop duplexing the data to staging data sets and duplex the coupling facility data to local storage buffers, because there is no longer a single point of failure in the volatile coupling facility.

If the coupling facility changes from non-volatile to volatile, the coupling facility structure is rebuilt. When the rebuild completes, system logger re-evaluates each connection for a single point of failure and whether to use staging data sets.

Use STG_DUPLEX(YES) DUPLEXMODE=(COND) when you cannot afford any data loss. It is a flexible way to protect your application against a single point of failure on a connection by connection basis. With DUPLEXMODE=COND, duplexing to staging data sets is only used when needed for a particular connection. Connections that do not have a single point of failure can exploit the performance benefits of backing up log data to local storage buffers when duplexing is not necessary.

- DUPLEXMODE(UNCOND) specifies that system logger is to unconditionally duplex coupling facility data for log stream connectors to DASD staging data sets. Use this option when you always want to use staging data sets, regardless of whether the connection contains a single point of failure or not. The only possible exception to logger unconditionally using staging data sets for this specification is when LOGGERDUPLEX(COND) is also specified for the log stream. For this case, if the structure is undergoing a system-managed duplexing rebuild into two failure independent structure instances, and if there is no other single point of failure between logger and the structure, logger will not provide any log data duplexing.

Example - Using staging data sets when the coupling facility shares a CPC with a system

In Figure 67 on page 285, a log stream has been defined with STG_DUPLEX=YES DUPLEXMODE=COND. There are two systems connected to the log stream. System 1, SYS1, shares the CPC with the coupling facility, so the data that SYS1 writes to the coupling facility is duplexed to a staging data set because there is a single point of failure between the system and the coupling facility. For example, a machine error might require a reset of CPC1, which would result in the loss of both the coupling facility and SYS1 with its local storage buffer copy of the coupling facility log data if it was not duplexed to staging data sets.

System 2, SYS2, is in a different CPC from the coupling facility it writes to, so the backup copy of coupling facility log data for that system connection is maintained in a local storage buffer within SYS2. If SYS2 should fail, its data is still in the coupling facility list structure.

If the coupling facility fails, SYS2 still has the coupling facility log data for this connection in the local storage buffer. The local storage buffer copy of the coupling facility data can be used to recreate the log stream in another coupling facility list structure.

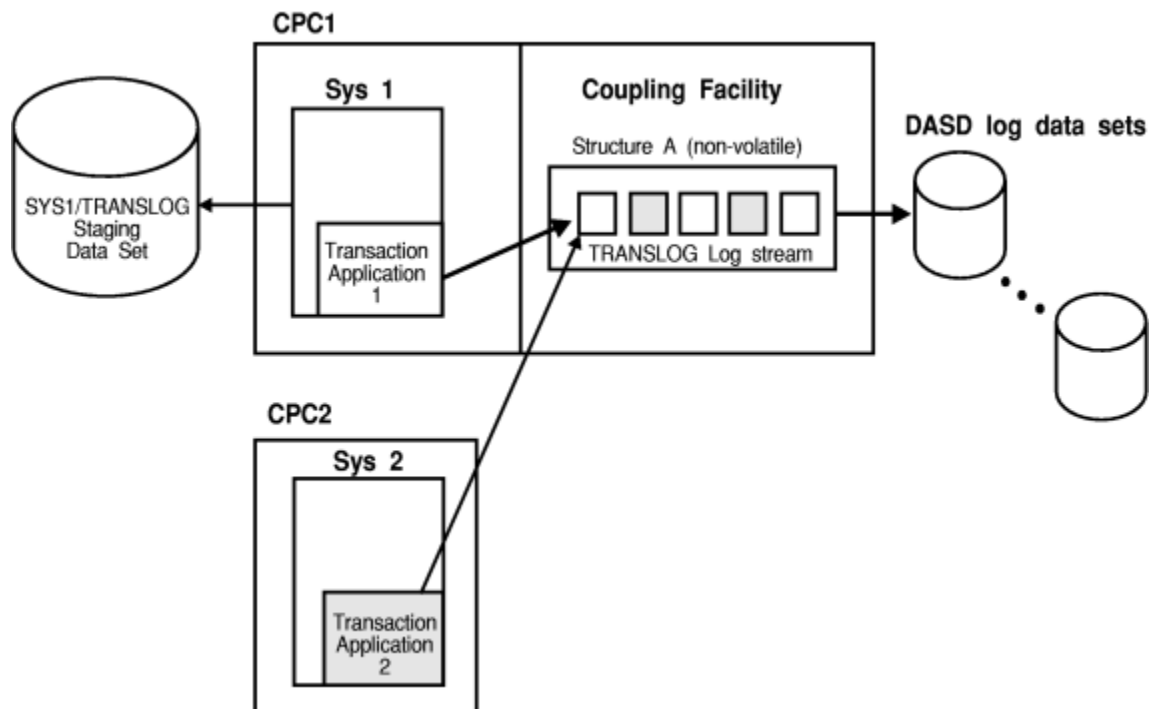


Figure 67. Example: Using Staging Data Sets When the Coupling Facility Shares a CPC with a System

If the installation had specified STG_DUPLEX(YES) DUPLEXMODE(UNCOND), both systems would duplex to staging data sets, even though there is no single point of failure for SYS2.

If the installation had specified STG_DUPLEX(NO), neither system would duplex to staging data sets (under normal circumstances).

Example - Using staging data sets when the coupling facility is volatile

In Figure 68 on page 286, a log stream was defined in the LOGR policy with STG_DUPLEX(YES) DUPLEXMODE(COND). Two systems are connected to the log stream. The coupling facility structure associated with the log stream is in a stand-alone CPC. The coupling facility for the structure is volatile, therefore both systems will duplex the data to staging data sets because there might be a single point of failure for both systems.

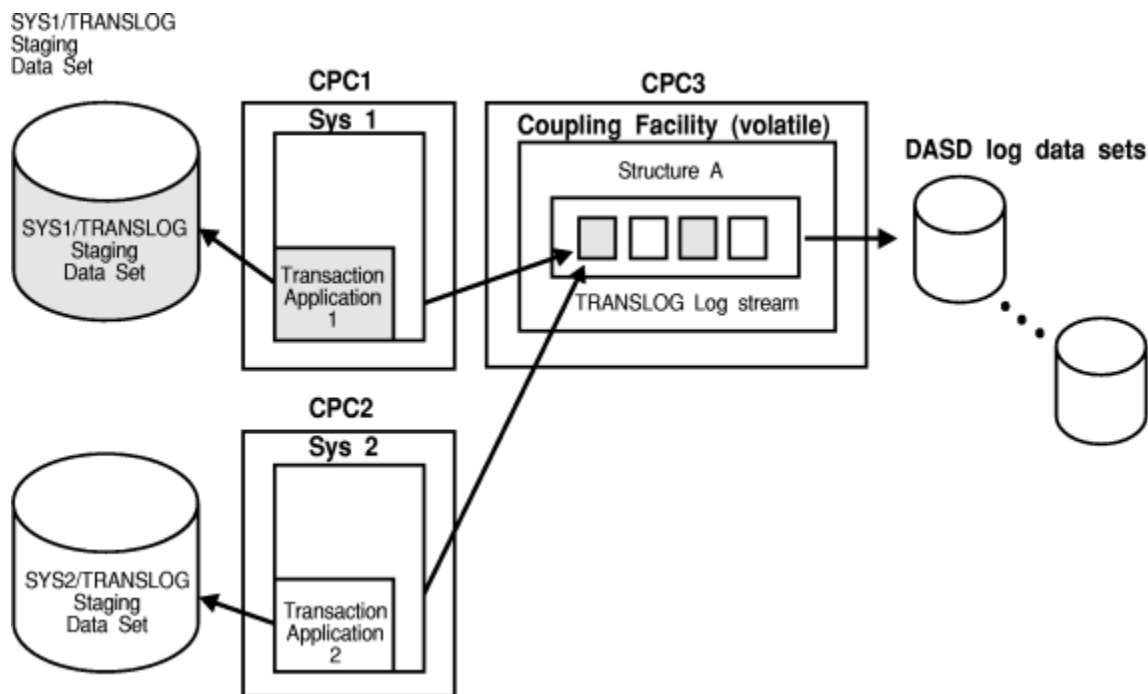


Figure 68. Example: Using Staging Data Sets When the Coupling Facility is Volatile

If the coupling facility was non-volatile, neither system would duplex their coupling facility data to staging data sets, but would maintain local buffers on each system.

If the installation had specified `STG_DUPLEX(YES) DUPLEXMODE(UNCOND)`, both systems would duplex data to staging data sets, regardless of the volatility of the coupling facility.

If the installation had specified `STG_DUPLEX(NO)`, neither system would duplex to staging data sets (under normal circumstances).

Logger and coupling facility duplexing combinations

In addition to the types of duplexing methods that Logger provides, coupling facility structures can also be duplexed automatically. System-managed duplexing rebuild provides the capabilities of keeping two instances of a coupling facility structure, one a duplicate of the other. When there are two duplicate structure instances, the structure is considered to be in duplex-mode. If there is only one structure instance, the structure is considered to be in simplex-mode.

The Logger log stream `LOGGERDUPLEX` parameter and your parallel sysplex configuration will determine whether Logger will duplex the log data or whether system-managed duplexing rebuild will occur using duplex-mode coupling facility structures.

The following factors determine the type and whether Logger duplexing will occur with respect to the coupling facility structure environments:

- The coupling facility's volatility state
- The simplex-mode or duplex-mode of the structure
- The connection relationship between the structure instances
- Their composite connection relationship with the connecting system
- The Logger log stream duplexing options (new keyword `LOGGERDUPLEX` and existing keywords `STG_DUPLEX` and `DUPLEXMODE`).

Table 23 on page 287 illustrates the type of log data duplexing that will be in effect based on the combination of these factors.

Table 23. Logger and System-Managed Duplexing Rebuild Combinations				
Structure Mode / Case #	Logger Duplex	STG_DUPLEX (NO)	STG_DUPLEX (YES)	
			DUPLEXMODE (COND)	DUPLEXMODE (UNCOND)
SIMPLEX				
Case 1	UNCOND / COND	Local Buffers	Staging Data Set	Staging Data Set
Case 2	UNCOND / COND	Local Buffers	Local Buffers	Staging Data Set
DUPLEX				
Case 3	UNCOND / COND	Structure, Local Buffers	Structure, Staging Data Set	Structure, Staging Data Set
Case 4	UNCOND / COND	Structure, Local Buffers	Structure, Local Buffers	Structure, Staging Data Set
Case 5a	UNCOND	Structure, Local Buffers	Structure, Local Buffers	Structure, Staging Data Set
Case 5b	COND	Structure	Structure	Structure

Note:

1. In Table 23 on page 287, the Mode / Case # column is defined as follows:

simplex-mode

Only one structure instance (system-managed duplexing is not in effect):

- Case 1 — a failure dependent connection between the connecting system and the structure, or the structure coupling facility state is volatile.
- Case 2 — a failure independent connection between the connecting system and the structure, and the structure coupling facility state is non-volatile.

duplex-mode

There are two structure instances (system-managed duplexing is in effect):

- Case 3 — a failure dependent connection between the connecting system and the composite structure view, or the structure coupling facility state is volatile.
- Case 4 — a failure independent connection between the connecting system and the composite structure view, and a failure dependent relationship between the two structure instances, and the structure coupling facility state is non-volatile.
- Case 5 — a failure independent connection between the connecting system and the composite structure view, and a failure independent relationship between the two structure instances, and the structure coupling facility state is non-volatile.

2. The characterizations in Table 23 on page 287 indicate what type of duplexing would be in effect for the particular case and parameter specifications:

local buffers

Logger is duplexing data in data spaces associated with the IXGLOGR address space.

staging data set

Logger is duplexing the log data in a staging data set associated with the system's connection to the log stream and structure.

structure

The log stream data is duplexed in two instances of the structure. Logger is not duplexing the log data because the configuration and duplexing options indicate it is not necessary.

The new environments that can exist when the coupling facility structure is in duplex-mode offer additional log data duplexing and recovery opportunities.

- Case 3 demonstrates that even though the structure is in duplex-mode the composite view of the structures to the connecting system is failure-dependent (or in a volatile state). There is no

recoverability advantage with the two structure instances since there is still a single point of failure dependency. This environment is treated in a similar manner as in the simplex-mode Case 1 above.

- Case 4 demonstrates that even though the composite view of the structures to the connecting system is failure-independent (and non-volatile), the two structure instances are failure-dependent so Logger would continue to provide its own duplexing. Since a single point of failure exists for the two structure instances, Logger will continue to use its own duplexing method for log data recovery. This environment is treated in a similar manner as in the simplex-mode Case 2 above.
- Case 5 demonstrates that given the appropriate failure isolated relationship between the connecting system as well as between the two structure instances, the duplexing and recovery of the log data can be handled using either the duplex-mode structure or from Logger's duplexed copy.

See [“System logger recovery” on page 300](#) for additional details on how Logger handles recovery of log data for the different failure conditions, along with the system-managed duplexing rebuild and single point of failure transitions based on the types of log data duplexing.

Migration steps to enable system-managed duplexing of logger structures

There are system logger actions that you must take to enable system-managed duplexing of Logger structures, in addition to the considerations and steps described in [“An installation guide to duplexing rebuild” on page 69](#) and [“Migration steps” on page 71](#).

Base requirements

To allow system-managed duplexing by XES on a Logger structure, the LOGR couple data set used in the sysplex must be at the HBB7705 format level (this requires all systems be at the z/OS 1.2 level or greater), so that Logger can newly connect to the structure specifying IXLCONN ALLOWAUTO=YES. See [“LOGR parameters for format utility \(sysplex scope\)” on page 320](#) for bringing in a LOGR couple data set at the z/OS 1.2 format level (HBB7705).

Activation steps when there are no current logger structure connections

Once the base requirements are met and a new structure allocation occurs, then system-managed duplexing can be used for the structure.

Activation steps when there are existing logger structure connections

In addition to the base requirements, action is required for logger structures to become eligible for system-managed duplexing when there are existing structure connections. Note that simply causing a rebuild for logger structures will not satisfy the configuration or environment necessary for migrating existing structures to becoming duplexed.

Additionally, it is not necessary to take the entire sysplex down at one time in order to switch to duplex mode. Only the systems that have connections to the structures of interest will require action to be taken, causing Logger to newly connect to the structure with the appropriate options so system-managed duplexed structures can be used.

Things to consider when logger structure connections exist

- Before using a primary LOGR couple data set at HBB7705, formatted with ITEM NAME(SMDUPLEX) NUMBER(1), ensure that each system has specified ALLOWAUTO(NO) during the initial connection to the structure when using IXLCONN.
- The Logger connection type doesn't change until the system newly reconnects to the structure (not on IXLCONN REBUILD connection requests).
- After the LOGR couple data set formatted at HBB7705 is brought in as the primary couple data set, any new Logger system connections to the structure will use the ALLOWAUTO(YES) option.
- If there are still outstanding IXLCONN ALLOWAUTO(NO) connections to the structure from any system in the sysplex, then XES will not enable the system-managed duplexing.

- If the logstream application or subsystem and Logger on each system is caused to disconnect and then (newly) reconnect to the Logger structure one at a time, all the new structure connections will then be done with IXLCONN ALLOWAUTO(YES).
- When all the systems connected to a structure use the ALLOWAUTO(YES) option, system-managed duplexing options are now activated.

See [“Migration steps” on page 71](#) for migration steps to allow system-managed duplexing rebuild in a sysplex. See [“LOGR couple data set versioning — new format levels” on page 321](#) for migration steps to use new format levels of the LOGR couple data set.

Associating a log stream with a resource manager

This step applies to both coupling facility and DASD-only log streams.

If an application is using a resource manager, you can associate a log stream with a resource manager using the RMNAME parameter in the LOGR policy. You can specify **one** resource manager name for a log stream in the log stream definition. A resource manager is an application you can write to manage resources and processing for a log stream. See the system logger chapter in *z/OS MVS Programming: Authorized Assembler Services Guide* for more details concerning resource managers.

Deleting log streams from the LOGR, LOGRY or LOGRZ policy

To delete a log stream you must delete it from the LOGR policy. To accomplish this you can use either:

- The IXCMIAPU utility for system logger, with DELETE LOGSTREAM NAME(*log_stream_name*) parameters.
- The IXGINVNT service, with the REQUEST=DELETE TYPE=LOGSTREAM parameters.
- The SETLOGR FORCE,DELETE command with the LSN=name parameter.

When using a system logger single-system scope LOGRY or LOGRZ policy, you can use one of the above approaches to delete a log stream by submitting the request on the system that is using the respective single-system scope CDS type policy.

If there are active connections to the log stream you are trying to delete, or failed connections that cannot be recovered, system logger will not let you delete the log stream. The requestor of the delete will be notified if there are any connectors remaining to that log stream.

You can find out if a log stream has existing connections using the DISPLAY LOGGER command or by using the LIST parameter on the IXCMIAPU utility for system logger. For example, to get information about connectors to a log stream in the SYSPRINT output, use LIST LOGSTREAM NAME(*log_stream_name*) DETAIL(YES) on the IXCMIAPU utility for system logger. For the LIST parameter of IXCMIAPU, see [Chapter 12, “Administrative data utility,” on page 335](#).

If the log stream you wish to delete still has one or more active connectors, you can do one of the following:

- Keep the connection, canceling deletion of the log stream.
- Cause the application to disconnect from the log stream or force the log stream to be disconnected using the SETLOGR command, and then delete the log stream from the LOGR policy.

If a log stream delete attempt fails using the IXCMIAPU utility, or the IXGINVNT service, and there are no longer active connections to the log stream, the SETLOGR FORCE,DELETE command may allow you to delete the log stream.

Define the coupling facility structures attributes in the CFRM function couple data set

This step applies to coupling facility log streams only. DASD-only log streams can omit this step.

Do the following:

- **Determine the attributes needed to define coupling facility structures used by system logger applications.**

See [“Planning a coupling facility policy”](#) on page 49 to plan updates to the CFRM policy for system logger applications. When planning preference and exclusion list specifications, keep in mind that system logger requires a minimum CFLEVEL. See [“Understand the requirements for system logger”](#) on page 244 for specific requirements.

The ALLOWAUTOALT(NO|YES) parameter on a structure definition in the CFRM policy may affect structures used by system logger. Since logger connects to a coupling facility list structure using ALLOWALTER(YES), if you specify ALLOWAUTOALT(YES) on the list structure definition in the CFRM policy, Cross System Extended Services (XES) may automatically alter the size and ratio attributes of the structure as needed. If your coupling facility is size constrained and you must free up some space, you should probably consider other structures for automatic altering first. If you must allow automatic altering of the logger list structure, IBM suggests that you also specify a minimum size for the logger list structure with the MINSIZE parameter, to ensure that logger always has enough space to do its work.

The effects associated with the automatic altering of a list structure in use by logger can vary depending on the number of log streams connected to the structure. Consider the following cases:

- Case 1: A single log stream is connected to the list structure.

Depending on how your applications use the log stream, you can find that there are long periods when the log stream is relatively empty, followed by peaks of system activity when the log stream quickly approaches its high threshold. If you have specified ALLOWAUTOALT(YES), XES may decide to alter your structure size to a smaller size during a low usage period. When a period of peak usage occurs, write requests to this now smaller structure area for this log stream could result in a structure full condition, meaning there is no more space available in the structure for this log stream. You can avoid this by specifying a sufficient MINSIZE for the list structure.

- Case 2: Multiple log streams are connected to the list structure.

In this case, since there is more than one log stream mapped and connected to the list structure, the overall use of the structure space may not be indicative of how any one log stream is performing. For example, assume one log stream in the structure is using most, but not all, of its share of the list structure space, and a second log stream in the same structure is using very little of its share of the list structure space. If automatic monitoring and altering processing is allowed it could reduce this structure's size because the overall use of the list structure space is relatively low. This size reduction could cause the log stream that was using a large portion of its available space to now exceed its available space. Subsequent write requests to this log stream would result in a structure full condition, meaning there is no more space available in the structure for this log stream.

The “structure full monitoring” function became available in OS/390 V2R9. See [“Monitoring structure utilization”](#) on page 60.

To allow installations to control the monitor level, a new parameter was added to the CFRM policy, FULLTHRESHOLD. Use this parameter to set the utilization value to which XCF is to monitor the structure, with the default value being 80%. Setting FULLTHRESHOLD(0) disables the monitor for that structure. For system logger structures, it is advisable to set FULLTHRESHOLD to some value higher than the HIGHOFFLOAD log stream parameter. XCF, when monitoring the number of ENTRIES and ELEMENTS, will issue a warning message (IXC585E) when the value reaches the FULLTHRESHOLD value. System logger will not perform offload processing until the ELEMENTS value is at the HIGHOFFLOAD value. Note that the HIGHOFFLOAD parameter is on a per log stream basis, while the FULLTHRESHOLD parameter is on a structure basis. For this reason, try to group log streams together that have similar HIGHOFFLOAD values.

- **Add information to the CFRM function couple data set about the desired attributes of the coupling facility list structures for system logger applications.** To use the IXCMIAPU utility to update the CFRM policy for logger information, see [“CFRM parameters for administrative data utility”](#) on page 346.

Note that each coupling facility structure that you define in the LOGR policy must match those defined in the CFRM policy. See [“CFRM parameters for administrative data utility”](#) on page 346 for using IXCMIAPU to update the CFRM policy.

- **Ensure adequate dump space is specified in a CF definition in the CFRM policy that could contain list structures that will be used by the system logger.**

See [“Determining the amount of coupling facility dump space” on page 65](#) for setting the DUMPSPACE parameter.

- **Activate the CFRM function couple data set using the SETXCF command.**

See [z/OS MVS System Commands](#) for the SETXCF command.

Activate the LOGR subsystem

The LOGR subsystem allows you to read log stream output without having to re-write your existing applications. For example, you might have existing applications that read logrec data in data set format and do not support the log stream format. The LOGR subsystem lets you access log stream data in data set format.

If your installation chooses to use the LOGR subsystem to read log stream output, you must activate it before using the log stream. Activate the LOGR subsystem using one of the following methods:

- In the IEFSSNxx parmlib member, add the following:

```
SUBSYS SUBNAME(LOGR) INITRTN(IXGSSINT)
```

Note that the new format for defining the LOGR subsystem is required (you cannot use the old format). Because IEFSSNxx parmlib members cannot contain the new and old format definitions in the same IEFSSNxx parmlib member, you might need to define a new parmlib member for the LOGR subsystem and concatenate it with other IEFSSNxx parmlib members.

- Use the SETSSI command to add the subsystem dynamically:

```
SETSSI ADD,SUBNAME=LOGR,INITRTN=IXGSSINT
```

The value specified for SUBNAME is defined by the installation; it does not have to be LOGR. Make sure, however, that the subsystem name specified in the SUBNAME parameter matches the corresponding name in the SUBSYS option in the JCL.

Note that you cannot issue either the SETSSI ACTIVATE or SETSSI DEACTIVATE command for the LOGR subsystem.

For more information about the topics described in this section, see the following references:

- See [z/OS MVS Programming: Assembler Services Guide](#) for information about using the LOGR subsystem.
- See [z/OS MVS Initialization and Tuning Reference](#) for information about the IEFSSNxx parmlib member.
- See [z/OS MVS System Commands](#) for information about the SETSSI command.
- See [z/OS MVS Installation Exits](#) for information about the LOGR subsystem exit.
- See [“Authorization for system logger applications” on page 275](#) for additional information on allowing authorization for system logger applications.

Deleting log data and log data sets

In general, applications should handle deletion of log data from a log stream. If the application does not manage deletion, the installation might be able to use the retention period and automatic deletion functions in system logger to manage the amount of log data you keep in each log stream. See [“Managing log data: How much? For how long?” on page 272](#) for the retention period and automatic deletion. You can also increase the amount of space available for DASD log data sets using the DSEXTENT parameter in the LOGR couple data set. See [“Increasing the space available for DASD log data sets” on page 262](#).

If for some reason, your installation wants to delete log data, first carefully evaluate the needs of application or applications writing data to the log stream. For some log streams, deleting log data is not a problem, while for others it could be disastrous.

This topic covers the following:

- [“Deleting log data” on page 292.](#)
- [“Deleting log data sets” on page 292.](#)
- [“When is my log data or log data set physically deleted?” on page 292.](#) System logger **does not** delete log data or log data sets immediately when you mark them for deletion.
- [“Finding and deleting orphaned log data sets” on page 293.](#) Deletion of log data sets may result in errors that leave orphaned log data sets consuming storage.

Deleting log data

Use one of the following methods to delete log data:

- Use the IXGDELETE service in an application to delete data from the log stream as soon as the data is no longer needed. This will help to manage log data set storage consumption by making sure log data sets are deleted as soon as possible. See [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#) for using the IXGDELETE service.

Use RETPD / AUTODELETE logstream options when new data sets are acquired during offload processing. If you specify the RETPD and AUTODELETE parameters to set a retention period and automatic deletion policy for a log stream, the data set becomes eligible for deletion once the retention period has expired for all the log data in the log data set.

- For logrec log stream data, run EREP using the LOGR subsystem with the delete option. See program IFBEREPS in SYS1.SAMPLIB for examples of archiving and deleting logrec log data sets. See [z/OS MVS Diagnosis: Tools and Service Aids](#) for using EREP.
- For OPERLOG, use the IEAMDBLG program in SYS1.SAMPLIB with the DELETE parameter.
- For the CICS log manager, CICS automatically deletes data in a CICS system log. For CICS general logs, use the RETPD and AUTODELETE parameters in the log stream definition to manage the amount of log data automatically. You can also use CICS/VR to archive or delete CICS log data.

Note that using DFHSM to migrate log stream data sets to tape **does not** solve the data set directory full condition because the migrated data sets will still count toward the data set directory entry limit. Deleting DASD log stream data sets (using a TSO/E DELETE command, for example) also **does not** solve the data set directory full condition.

Deleting log data sets

To delete a log data set use one of the methods for making the offload data set eligible for deletion as described in [“Deleting log data” on page 292](#), then either wait or cause a logger offload and data set switch to occur or cause the application to disconnect from the logstream.

Disconnecting from the logstream may cause offload processing to occur. Offload processing may cause a data set switch to occur. Data set switch processing may decide to delete log data sets if they meet all of the eligibility requirements.

The deletion of the log stream definition from the LOGR policy will cause the deletion of all the offload data sets associated with the deleted log stream. See [“Deleting log streams from the LOGR, LOGRY or LOGRZ policy” on page 289](#) and [“When is my log data or log data set physically deleted?” on page 292](#).

Note that IBM **does not** suggest other methods for deleting data sets, such as the TSO/E DELETE command, because system logger cannot perform any clean-up and the deleted data sets will still count toward the data set directory limit.

When is my log data or log data set physically deleted?

After log data is marked for deletion, system logger does not physically delete the log data or log data set until an offload requires a new data set to be allocated. This means that there will often be a delay before eligible log data sets are physically deleted, since offloading will not occur until the high threshold associated with the log stream is reached. If off-loads are relatively infrequent, then there may be a considerable delay before log data sets that are eligible for deletion are actually deleted.

Allocation errors can also delay log data set deletion. If system logger cannot delete a log data set because it is currently allocated, it will attempt to delete the log data set on a subsequent offload when delete processing is again performed. This situation can also delay the deletion of a log data set since system logger may attempt to delete an eligible log data set several times before it actually succeeds. See [“Finding and deleting orphaned log data sets”](#) on page 293.

If you specify the RETPD and AUTODELETE parameters to set a retention period and automatic deletion policy for a log stream, the data set becomes eligible for deletion once the retention period has expired for all the log data in the log data set. System logger deletes the data on a log data set basis when the log data set is full, is no longer in use as the current log data set for the log stream, and is not allocated to any system in the sysplex the next time a data set switch occurs.

Finding and deleting orphaned log data sets

When you delete a log data set, allocation errors may result in an “orphaned” log data set — a log data set that should have been deleted but was not. When a log data set is orphaned, system logger issues message IXG252I. System logger frees up the data set's entry in the directory extent in this case, but the data set still consumes storage. If an allocation error occurs when you delete a log stream or when system logger is deleting a log data set, you will need to manually delete the resulting unnecessary log data sets. The orphaned data sets should be deleted using the IDCAMS delete command. Note that when you delete a log stream by deleting it from the LOGR policy, system logger deletes all the log data sets associated with the log stream that have entries in the logger directory. If a data set was orphaned or simply matches the log stream naming convention but is not in the logger directory, then system logger will NOT attempt to delete the data set when the log stream is deleted.

If you suspect you might have possible orphaned data sets that may have previously been associated with a log stream, use `LIST LOGSTREAM NAME(log_stream_name) DETAIL(YES)` on the IXCMIAPU utility for DATA TYPE(LOGR).

If you rename log streams using the `UPDATE LOGSTREAM NEWSTREAMNAME` keyword, offload data sets from a different log stream instance with the same original name might appear as orphaned data sets when viewing the LIST output. Before taking any action on orphaned data sets, determine if any of them are associated with an existing log stream that had been renamed.

See the topic in the output report entitled "POSSIBLE ORPHANED LOG STREAM DATA SETS:" to find any data sets that match the log stream naming convention but are not part of system logger's data set directory for the log stream. For the LIST parameter of IXCMIAPU, see [Chapter 12, “Administrative data utility,”](#) on page 335.

Upgrading an existing log stream configuration

You can upgrade a DASD-only log stream to use coupling facility storage, making it a coupling facility log stream, or you can update an existing structure-based log stream to use a different coupling facility structure.

Changing the GROUP value

When updating an existing log stream configuration, the group attribute must be taken into account. The following sections contain examples of changing the GROUP value for existing log streams and structures:

Example 1—Change the GROUP value of a log stream that is the only log stream defined to a structure

The following steps show an example of changing the GROUP value of a log stream that is the only log stream defined to a structure. Assume that there is log stream LS1 in the structure STRUCA with the value of GROUP(TEST).

1. Disconnect all applications to log stream LS1
2. Use utility IXCMIAPU to update LS1:

```
UPDATE LOGSTREAM NAME(LS1) GROUP(PRODUCTION)
```

This assigns structure STRUCA with the value of GROUP(PRODUCTION) because it changes the value of log stream LS1 to GROUP(PRODUCTION).

Example 2 - Change the GROUP value for all the log streams in a structure

The following steps show an example of changing the GROUP value for all the log streams in a structure. Assume that there are log streams LS1, LS2, LS3, LS4, and LS5 in the structure STRUCA with the log streams all defined GROUP(TEST).

1. Disconnect all applications for the log streams named above
2. Use IXCMIAPU to define a new structure STRUCB:

```
DEFINE STRUCTURE NAME(STRUCB) LOGSNUM(5)
```

There will be no GROUP value in the command D LOGGER,STR,STRN=STRUCB. Note that you only need to define this structure to the LOGR CDS and you do not need to define it to the CFRM policy. The LOGSNUM attribute should be at least equal to the number of log streams you are moving.

3. Use IXCMIAPU to update the log stream LS1:

```
UPDATE LOGSTREAM NAME(LS1) STRUCTNAME(STRUCB) GROUP(PRODUCTION)
```

This step assigns STRUCB with GROUP(PRODUCTION) and changes log stream LS1 to GROUP(PRODUCTION).

4. Do the same for log streams LS2, LS3, LS4, and LS5 by specifying STRUCTNAME(STRUCB) GROUP(PRODUCTION) on each IXCMIAPU UPDATE.

When all log streams LS1, LS2, LS3, LS4, and LS5 in STRUCB are defined with the desired attribute of GROUP(PRODUCTION), the original structure STRUCA should have no log streams assigned to it. This allows the original log streams to be brought back into STRUCA with the value of GROUP(PRODUCTION).

5. Use IXCMIAPU to update LS1 in STRUCA:

```
UPDATE LOGSTREAM NAME(LS1) STRUCTNAME(STRUCA)
```

This step assigns STRUCA with the PRODUCTION value because log stream LS1 is already defined to GROUP(PRODUCTION).

6. Do the same for log streams LS2, LS3, LS4, and LS5 in STRUCB by specifying STRUCTNAME(STRUCA). And then log streams LS1, LS2, LS3, LS4, and LS5 are assigned to structure STRUCA with the value of GROUP(PRODUCTION).

Example 3 - Have log streams with both values of GROUP(PRODUCTION) and GROUP(TEST)

To have some log streams with both values of GROUP(PRODUCTION) and GROUP(TEST), the approach is significantly different from the previous ones. There must be two distinct structures - one to hold one or more GROUP(PRODUCTION) log streams and the other to hold one or more GROUP(TEST) log streams. There should be active connections among all of the log streams, and the two structures must not only be known to LOGGER but to XES as well. Therefore, the new structure must be defined and activated through the CFRM policy.

Assume that there are log streams LS1, LS2, LS3, LS4, and LS5 with the value of GROUP(TEST) assigned to the structure STRUCA and assume that LS1, LS3, and LS5 need to be with the value of GROUP(PRODUCTION) while LS2 and LS4 remain with the value of GROUP(TEST).

1. Use IXCMIAPU DEFINE to create a new structure, STRUCB.

```
DEFINE STRUCTURE NAME(STRUCB) LOGSNUM(5)
```

2. Disconnect all applications for log streams LS1, LS3 and LS5.
3. Use IXCMIAPU UPDATE to assign LS1 to STRUCB with the attribute of GROUP(PRODUCTION):

```
UPDATE LOGSTREAM NAME(LS1) STRUCTNAME(STRUCB) GROUP(PRODUCTION)
```

4. Repeat this IXCMIAPU UPDATE for log streams LS3 and LS5 with GROUP(PRODUCTION) STRUCTNAME(STRUCB).

There should now be logger structure STRUCA with log streams LS2 and LS4 with the value of GROUP(TEST) and a distinct second structure STRUCB with log streams LS1, LS3, and LS5 with the value of GROUP(PRODUCTION).

5. Define the new structure, STRUCB, to XES with an update to the CFRM policy.
6. Once defined to XES, activate the new CFRM policy before attempting to connect to log streams on the structure.

Now there are two distinct and active structures: STRUCA with log streams LS2 and LS4 that are with the GROUP(TEST) value and STRUCB with log streams LS1, LS3, and LS5 that are with the GROUP(PRODUCTION) value.

Upgrading a log stream from DASD-only to coupling facility

To upgrade a log stream from DASD-only to coupling facility based, associate a structure with the log stream by updating the log stream definition in the LOGR policy. Use the IXGINVNT service or the IXCMIAPU utility, and on an UPDATE request specify a coupling facility structure name on the STRUCTNAME parameter. All connectors to the DASD-only log stream must be disconnected in order to make the upgrade. No failed or active connections can exist for this log stream.

Before upgrading a DASD-only log stream, read the following topics:

- [“Plan for a DASD-only log stream upgrade” on page 295](#)
- [“Match the MAXBUFSIZE values for a DASD-only log stream upgrade” on page 295](#)
- [“Staging data sets and upgrading a DASD-only log stream” on page 296](#)

Plan for a DASD-only log stream upgrade

Before upgrading a DASD-only log stream, make sure that you have prepared adequately for the new coupling facility log stream. You need to be familiar with the coupling facility log stream requirements and planning steps covered in [“Preparing to use system logger applications” on page 243](#). For example, the structure specified on the log stream definition must be defined in the LOGR policy. And before you can connect to the upgraded log stream or issue other system logger services, you must also specify the structure in the CFRM policy (see [“Add information about log streams and coupling facility structures to the LOGR policy” on page 280](#)). In addition, the naming convention system logger uses to allocate staging data sets is different for coupling facility log streams and DASD-only log streams. See [“Naming conventions for the log stream and DASD data sets” on page 258](#).

Match the MAXBUFSIZE values for a DASD-only log stream upgrade

When you upgrade a log stream from DASD-only to coupling facility based, make sure that the structure you specify on the UPDATE request has a MAXBUFSIZE value that is equal to or greater than the MAXBUFSIZE value currently defined to the DASD-only log stream. For example, if you decide to upgrade a DASD-only log stream that has a MAXBUFSIZE of 65,532 bytes (which is the default), you must associate it with a structure that has a MAXBUFSIZE of 65,532 bytes.

If the MAXBUFSIZE value defined for the structure is smaller than the MAXBUFSIZE value for the DASD-only log stream, system logger ends the UPDATE request with an X'083C' reason code. To avoid this error, do one of the following before upgrading from a DASD-only log stream to a coupling facility based one:

- When specifying log stream definitions for DASD-only log streams, plan ahead for possible upgrades by matching the MAXBUFSIZE value for the DASD-only log stream with the MAXBUFSIZE value for the

structure you might assign it to on an upgrade request. The MAXBUFSIZE value on the DASD-only log stream definition must be the same size as or smaller than the MAXBUFSIZE value for the structure.

- On the UPDATE request to upgrade a DASD-only log stream, specify a structure with a MAXBUFSIZE value that is as large or larger than the DASD-only log stream MAXBUFSIZE value.

Note that you **cannot** issue an UPDATE request to make a MAXBUFSIZE smaller on a DASD-only log stream definition. You also cannot specify the MAXBUFSIZE parameter on an UPDATE request for a structure definition.

Staging data sets and upgrading a DASD-only log stream

A DASD-only log stream automatically duplexes log data to DASD staging data sets. When you upgrade a DASD-only log stream to a coupling facility based log stream, you will still get duplexing to staging data sets **unless you specifically specify otherwise**. You must specify STG_DUPLEX(NO) when you upgrade on the UPDATE LOGSTREAM request to get a coupling facility log stream that does duplexes to local storage buffers rather than to DASD staging data sets. For example, if you do not want to use staging data sets after the upgrade, specify STG_DUPLEX(NO) on the UPDATE request. See [“Selecting a method of duplexing coupling facility log data” on page 283](#) for more duplexing options.

Updating a log stream to use a different coupling facility structure

With z/OS Release 3 and higher, it is possible to associate an existing structure-based log stream with a new coupling facility list structure without first deleting the log stream definition and then redefining it. The following requirements apply to this type of update.

- System-level requirements

The update to a new coupling facility structure can be accomplished with either the IXGINVNT service or the IXCMIAPU utility on systems at z/OS R3 and higher. In addition, the LOGR couple data set must be formatted at a z/OS R2 level or higher. If either of these system conditions are not met, the request to define the new structure will fail.

- Structure-definition requirements

There can be no connections (active or failed) in the sysplex to the existing structure-based log stream before defining the new coupling facility structure. The MAXBUFSIZE value for the new structure must be greater than or equal to the original structure's MAXBUFSIZE value or the system will fail the UPDATE request.

Updating a log stream's attributes

You can update certain attributes of a DASD-only or coupling facility log stream using either the IXGINVNT UPDATE service or the IXCMIAPU utility. The updates are immediately reflected in the log stream definition in the LOGR couple data set, but some remain pending and do not take effect until specific events occur during processing.

Most of the attributes can be updated while there is an outstanding (active or failed-persistent) connection to the log stream. In order to specify these updated attributes and have them become pending because of a log stream connection, the LOGR couple data set must be formatted at least at a certain format level. See [“LOGR parameters for format utility \(sysplex scope\)” on page 320](#) for more details.

Note: If this requirement is not met, the UPDATE request fails.

Pending updates will take effect at different points for each log stream attribute as follows:

- The RETPD and AUTODELETE attributes updates take effect when a new offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex.
- The LS-DATACCLASS, LS_SIZE, LS_MGMTCLASS, LS_STORCLASS, and LS_ALLOCAHEAD attribute updates will remain pending until a new offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a coupling facility log stream, the update also takes effect during the next structure rebuild (user-managed or system-managed).

- The LOWOFFLOAD, HIGHOFFLOAD, OFFLOADRECALL, STG_DATACLAS, STG_MGMTCLAS, STG_STORCLAS, STG_SIZE and WARNPRIMARY attribute updates remain pending until the subsequent first connection to the log stream in the sysplex. For a coupling facility log stream, the update also takes effect during the next structure rebuild (user-managed or system-managed). The STG_DATACLAS, STG_MGMTCLAS, STG_STORCLAS, and STG_SIZE attribute updates are also committed when a new staging data set needs to be allocated following one of the above conditions. In the case of a DASD-only logstream, the HIGHOFFLOAD, LOWOFFLOAD, OFFLOADRECALL, and WARNPRIMARY attribute updates are also committed on the next offload data set switch.
- The structure-based logstream attributes STG_DUPLEX, DUPLEXMODE, and LOGGERDUPLEX that are pending updates are committed only on the subsequent first connection or last disconnection to the logstream in the sysplex or until the next successful structure (user-managed) rebuild.
- The LOGGERDUPLEX, STG_DUPLEX, and DUPLEXMODE attribute updates remain pending until the subsequent first connection to the log stream in a sysplex or following a successful coupling facility user-managed structure rebuild.
- The MAXBUFSIZE attribute update remains pending until the subsequent first connection to the DASD-only log stream in the sysplex.
- The ZAI and ZAIDATA attributes can be updated while there is an outstanding connection to the log stream. For this case, the change will immediately be reflected in the log stream definition. The updated specification will take effect on the following logger events for this log stream:
 - On the subsequent first connection to the log stream on a system (z/OS image),
 - As a result of a SETLOGR FORCE,ZAICONN,LSN= command on the target system, or
 - As a result of a SETLOGR FORCE,ZAICONN,ALL command when there is a connection to this log stream currently using the ZAI=YES setting on the target system.

Note: Because the updated value can be used on a system by system basis, the installation should ensure that proper actions are taken on all the systems with connections to the log stream in order to make use of the current value.

For an overview of when the dynamic updates take effect for the system logger logstream attributes, see [Table 24 on page 297](#). In the table, “yes” indicates that the attribute takes effect during the activity listed in the column heading. “no” indicates that the attribute does not take effect during the activity listed in the column heading.

Table 24. System Logger logstream attribute dynamic update commit outline				
Logstream attribute	Last disconnect or first connect to logstream in sysplex	Switch to new offload data set	CF structure rebuild	Notes
RETPD	yes	yes	no	
AUTODELETE	yes	yes	no	
LS_SIZE	yes	yes	yes	
LS_DATACLAS	yes	yes	yes	
LS_MGMTCLAS	yes	yes	yes	
LS_STORCLAS	yes	yes	yes	
LS_ALLOCAHEAD	yes	yes	yes	
OFFLOADRECALL	yes	yes	yes	“1” on page 298
LOWOFFLOAD	yes	yes	yes	“1” on page 298
HIGHOFFLOAD	yes	yes	yes	“1” on page 298

Table 24. System Logger logstream attribute dynamic update commit outline (continued)

Logstream attribute	Last disconnect or first connect to logstream in sysplex	Switch to new offload data set	CF structure rebuild	Notes
WARNPRIMARY	yes	yes	yes	“1” on page 298
STG_SIZE	yes	no	yes	
STG_DATACLAS	yes	no	yes	
STG_MGMTCLAS	yes	no	yes	
STG_STORCLAS	yes	no	yes	
STG_DUPLEX	yes	no	yes	“2” on page 298 and “4” on page 298
DUPLEXMODE	yes	no	yes	“2” on page 298 and “4” on page 298
LOGGERDUPLEX	yes	no	yes	“2” on page 298 and “4” on page 298
MAXBUFSIZE	yes	no	n/a	“3” on page 298
ZAI	n/a	n/a	n/a	“5” on page 298
ZAIDATA	n/a	n/a	n/a	“5” on page 298

Notes for :

1. These attributes only take effect during a switch to new offload data set activity for DASD-only log streams. These attributes do not take effect at this point for coupling facility log streams.
2. Dynamic updates of the attribute is applicable only to coupling facility log streams.
3. The attribute is applicable only to DASD-only-based log streams.
4. The attribute pending update will take effect following a successful coupling facility user-managed structure rebuild. The attribute pending update will not take effect for coupling facility system-managed (for example: simplex to duplex mode) rebuilds nor for coupling facility structure duplex state changes (for example: duplex to simplex mode).
5. ZAI and ZAIDATA attribute updates are reflected in log stream definition regardless of log stream connection state.

A subset of the log stream attributes can only be updated while there are no connections (active or failed-persistent) to the log stream.

System logger log stream attribute update only when no connections:

- DESCRIPTION
- GROUP
- RMNAME
- STRUCTNAME

System logger log stream attributes: There are also a set of log stream attributes that cannot be updated, regardless of connections to the log stream. The following log stream attributes cannot be updated while the log stream is defined.

- EHLQ
- HLQ
- LIKE
- MODEL

Renaming a log stream dynamically

Using the NEWSTREAMNAME parameter on either the IXGINVNT REQUEST=UPDATE or IXCMIAPU UPDATE LOGSTREAM request, you can rename a log stream dynamically. Since many logging programs use a fixed log stream name, this function can be very useful for log stream recovery. For example, if a log stream fails the system may not be able to perform any logging for the log stream that has failed. Using the log stream rename function, you can rename the failing log stream, and then use IXGINVNT or IXCMIAPU to define a new log stream with the old name in order to start logging again. This function allows you to:

- Save the current data in a renamed log stream and get logging started quickly by defining a new log stream with the expected name.
- Let existing services/utilities access the data in the renamed log stream to reduce the effect of having some data missing from the log stream.
- Perform problem diagnosis, such as data missing conditions, on the original (renamed) log stream resources at the same time that new work continues.

You can use the NEWSTREAMNAME parameter only when there are no active connections to the log stream in the sysplex. You can, however, use NEWSTREAMNAME when there are failed persistent connections to the log stream. If you try to rename a log stream that has active connections, the system rejects the UPDATE request with return code 8, reason code X'0810'.

You must have an active primary TYPE=LOGR couple data formatted at an HBB7705 level or higher in order to specify the NEWSTREAMNAME parameter. Otherwise, the request will fail with a return code 8, and reason code X'0839'.

Updating a log stream with a new name will have the following results:

- Existing log stream offload data sets will remain unchanged. They are still part of the original log stream instance even though the log stream name is changed.
- Any subsequent log stream offload data sets newly created after a log stream is renamed will continue to use the original log stream name as the middle level qualifier in the construction of the new offload data sets.
- Any staging data sets that exist for log stream failed persistent connections will be renamed using the new log stream name as the middle level qualifier part of the construct. Logger will first rename the log stream resource name and commit the change to the LOGR couple data set (CDS). Relevant staging data sets will then be renamed.

The system will issue message IXG276I for each staging data set successfully renamed. In the event that Logger is unable to rename a staging data set, message IXG277E will be issued indicating the error condition and processing will continue. This will also result in return code 4, reason code X'0418' from the UPDATE request, but processing continues on the log stream update request. If you receive reason code X'0418', check for IXG251I hard-copy messages, and see the system programmer response for the message identifier that is included in message IXG251I. See [z/OS DFSMSdfp Advanced Services](#) and correct the condition that caused the error.

If a staging data set is migrated, the IXG251I messages may indicate that the data set is a NONVSAM type entry for the cluster. Migrated staging data sets for the log stream must first be recalled before submitting the NEWSTREAMNAME update request as Logger does not attempt to rename migrated data sets. Submit the necessary IDCAMS ALTER entryname NEWNAME() job to get the existing log stream staging data set name updated to match the new stream name change. This will need to be done before defining a new instance of a log stream that uses the same name as the log stream identified in this message. Failure to get the staging data set renamed correctly can result in a loss of data condition

when a connection occurs for the log stream that was renamed. If you cannot identify the problem source or correct the error, contact the IBM Support Center.

- The log stream name change has no impact on the HLQ or EHLQ value for the renamed log stream. **IBM suggests** that a unique HLQ or EHLQ be used when a new log stream is defined using the same original name as a previously defined and renamed log stream. This will cause Logger to create offload and staging data sets with names different than previously renamed log streams.
- A log stream that had been renamed can be renamed again with all the above rules and conditions still in effect.

System logger recovery

This topic explains how system logger handles errors or coupling facility structure state changes, how the errors might affect applications, and what action, if any, a system programmer might need to take. For coupling facility log streams, see also [“Plan your configuration for log data recovery” on page 251](#).

System logger performs recovery differently for DASD-only versus coupling facility log streams. Most of the information in this topic applies to coupling facility log streams only. For DASD-only log streams, see [“Recovery performed for DASD-only log streams” on page 300](#). Other recovery information pertinent to DASD-only log streams is noted under each topic.

Operations log and logrec application recovery

Each system logger application will issue messages to indicate that an error has occurred and explain it:

- **OPERLOG:** If operations log encounters an unrecoverable error, such as a damaged log stream or coupling facility structure, the system attempts to switch to a different hardcopy medium, such as SYSLOG or a device, based on what is defined on the HARDCOPY statement in the CONSOLxx parmlib member. (See [z/OS MVS Initialization and Tuning Reference](#) for more the CONSOLxx parmlib member.)

If an error occurs, the operations log issues message IEE316I indicating that OPERLOG has failed, the system where it has failed, and possibly some additional explanation of the problem. Message IEE316I might show some return and reason codes returned to operations log from system logger service; look these up in the [z/OS MVS Programming: Assembler Services Guide](#). Operations log issues message IEE710I showing the hardcopy medium that is active after OPERLOG fails.

- **Logrec log stream:** If logrec encounters a system logger error, logrec issues system message IFB100E explaining the problem. See the explanation of message IFB100E in [z/OS MVS System Messages, Vol 8 \(IEF-IGD\)](#).

If the error is recoverable, logrec resumes writing to the log stream. If the error is unrecoverable, an operator action might be required to switch hardcopy recording mediums or to resume writing to the log stream.

Recovery performed for DASD-only log streams

Like a coupling facility log stream, a DASD-only log stream is subject to system or system logger failure. A DASD-only log stream is not subject, however, to coupling facility or structure failures. When a failure occurs involving a DASD-only log stream, system logger releases the exclusive ENQ on the log stream name serializing the log stream for one system. No system-level or peer recovery is performed for a DASD-only log stream after a failure or as part of system initialization. System logger does not perform system-level recovery for a DASD-only log stream because data is already safeguarded on DASD staging data sets, a non-volatile medium. For a DASD-only log stream, offload of log data to DASD log data sets is not done as part of recovery processing for the same reason — log data is already on DASD staging data sets. Peer recovery is both unnecessary and not possible for a DASD-only log stream, because there are no peer systems connected to the log stream.

Recovery for a DASD-only log stream only takes place when an application reconnects to the log stream. As part of connect processing, system logger reads log data from the staging data set (associated with the last connection to the log stream) into the local storage buffers of the current connecting system. This allows the application to control recovery, by selecting which system they wish to have reconnect to the

log stream and when. Note that for another system to connect to the log stream and perform recovery, the staging data sets must reside on devices accessible by both systems.

When an MVS system fails

This topic applies to coupling facility log streams only. For DASD-only log streams, see [“Recovery performed for DASD-only log streams”](#) on page 300.

When a system fails, system logger tries to safeguard all the coupling facility log data for the failed system. This is done either by offloading it to DASD log data sets so that it is on a persistent media or by ensuring that the log data is secure in a persistent duplex-mode list structure.

Recovery processing for the failing system is done by a **peer connector**, which is another system in the sysplex with a connection to a coupling facility structure that the failing system was also connected to. Note that a peer connector need only be connected to the same coupling facility structure, not the same log stream. See [“Plan your configuration for log data recovery”](#) on page 251 for more information.

If there is no peer connection available to perform recovery for a failed system, recovery is delayed until either the failing system re-IPLs or another system connects to a log stream in the same coupling facility structure to which the failing system was connected.

If you have a failed system with no peer connectors (no other system is connected to the same coupling facility structure) and you want to minimize the time that the coupling facility data for the log stream remains unrecovered, IBM suggests one of the following procedures. Either of these procedures will ensure peer connectors for a system connection to a coupling facility structure so that recovery can be performed.

- Write a dummy application to run on another system that will connect to the same log stream or coupling facility structure to which the failed system was connected.
- Make sure that another log stream with connectors from other systems maps to the same coupling facility structure.

Recovery for a failing system in a single system sysplex

When a failing system was in a single system sysplex, there are no peer connectors existing to perform recovery for the failing system. In this case, the coupling facility resident log data is recovered when the failed system re-IPLs.

If the log data is still not recovered after IPL, do the following:

1. Delete the log stream from the LOGR data set; system logger will then write all the log stream data to DASD data sets.
2. If you receive a message that indicates that system logger cannot delete the log stream because it is 'in use', write a dummy application to run on another system that will connect to the same log stream or coupling facility structure to which the failed system was connected.

If the coupling facility that the system was connected to also fails, the system will lose its coupling facility resident log data unless staging data sets were in use. If staging data sets exist, system logger uses them to recover log data for the failing system. If there were no staging data sets in use when both the system and the coupling facility fails, any log streams affected are marked as damaged.

When a sysplex fails

This topic applies to coupling facility log streams only. For DASD-only log streams, see [“Recovery performed for DASD-only log streams”](#) on page 300.

When all the systems in a sysplex fail, there are no peer connectors to perform the recovery processing for the failing systems, which would consist of offloading the coupling facility data for log streams to DASD log data sets or ensuring the duplex-mode structure is still intact. Coupling facility resident log data continues to exist. Further recovery processing depends on whether the coupling facility also failed.

When the coupling facility survives the sysplex failure

When the coupling facility survives the sysplex failure, each system in the sysplex does the following:

1. Gets re-IPLed.
2. Identifies and reconnects to all the log streams to which it was connected before the failure occurs.
3. Performs recovery for the log streams (from the prior connections), which consists of offloading coupling facility log data to DASD log data sets or ensuring a duplex-mode structure is still intact.
4. Disconnects from the log streams that it reconnected to for recovery processing.
5. Performs recovery for any other log streams that were connected to the same coupling facility structures that the IPLing system had connections to before the failure.

Note that each IPLing system will perform recovery for log streams that it has a peer connection to, for as long as there is recovery to be done. Subsequent peer connectors IPLing will not duplicate recovery performed by other peer connectors.

Because system-level recovery for system logger is performed using the existing Coupling Facility structure information, you are using the correct LOGR couple data set when the systems are re-IPLed. See “LOGR couple data set use considerations” on page 325 for guidance on identifying and using an existing LOGR couple data set for use by the sysplex. Consider any environments that force the structure connections or allocations when you need to issue requests for a LOGR couple data set that makes use of structures that are connected to the system.

When the coupling facility fails with the sysplex

When the coupling facility fails at the same time as the sysplex, each system in the sysplex does the following:

1. Gets re-IPLed.
2. Identifies all the log streams to which it was connected before the failure occurred.
3. Performs recovery for log streams by trying to recover log data from staging data sets:
 - If each connector to a log stream was using staging data sets at the time of the failure, system logger recreates the log stream data from the staging data sets for each connection. If the recovery from staging data sets is successful, the log stream is considered recovered.

If system logger encounters an error while trying to recover log data from staging data sets, the log stream is marked as damaged. Subsequent IPLing systems will not attempt recovery from staging data sets because the log stream has been marked as damaged.

- If one or more connectors to a log stream did **not** use staging data sets while trying to recover log data from staging data sets, the log stream is marked as damaged.

Note that each IPLing system will perform recovery for the log streams that it has a peer connection to, for as long as there is recovery to be done. Subsequent peer connectors IPLing will not duplicate recovery performed by other peer connectors.

When the system logger address space fails or hangs

If system logger or an application using it appears to be hung or fails, there are several things that can be done to aid in identifying the problem. Topics following list some useful diagnostic procedures related to logger.

When the system logger address space hangs

The following commands can be useful as diagnostic aids when the system logger address space hangs. For a full description of these commands, see [z/OS MVS System Commands](#).

1. To display couple data sets in use for system logger:

```
D XCF,COUPLE,TYPE=LOGR
```

2. To display the status of a structure:

```
D XCF, STR,STRNAME=structurename
```

3. To display the status of the system logger address space (IXGLOGR) on this system:

```
D LOGGER,STATUS
```

4. To display the connection status of logstreams and jobs on this system:

```
D LOGGER,CONNECTION,...
```

5. To display logstream activity from a sysplex view (for example, from the LOGR couple data set):

```
D LOGGER,LOGSTREAM,...
```

6. To display structure activity from a sysplex view (for example, from the LOGR couple data set):

```
D LOGGER,STRUCTURE,...
```

7. To see if there is any contention:

```
D GRS,C
```

8. To see if there is a logger resource serialization contention:

```
D GRS,RES=(SYSZLOGR,*)
```

9. To see what units of work are blocking GRS-managed resources:

```
D GRS,ANALYZE,BLOCK,SYSTEM=*,JOBNAME=IXGLOGR
```

10. To see what units of work are waiting for ownership of GRS-managed resources:

```
D GRS,ANALYZE,WAIT,SYSTEM=*,JOBNAME=IXGLOGR
```

11. To see the dependencies between units of work and resources that are in contention:

```
D GRS,ANALYZE,DEPEND,SYSTEM=*,JOBNAME=IXGLOGR,DETAIL
```

12. Run the IXCMIAPU utility and specify the following:

```
DATA TYPE(LOGR) REPORT(YES)
LIST STRUCTURE NAME(structurename|*) DETAIL(YES)
LIST STRUCTURE NAME(structurename|*) DETAIL(YES)
```

See [Chapter 12, “Administrative data utility,”](#) on page 335 for a full description of the TYPE(LOGR) LIST options.

If there is a problem with a single application, attempt to stop and restart the application before stopping logger. If stopping the applications that have connections to log streams related to the difficulty, it may be necessary to force a log stream to be disconnected from a system using the SETLOGR FORCE,DISCONNECT command. See *z/OS MVS System Commands*.

If there is a problem with a structure, a rebuild may help resolve the problem. Issue the following command:

```
SETXCF START,REBUILD,STRNAME=structurename
```

If there is no logger (LOGR) couple data set, issue the following command to define the logger couple data set(s):

```
SETXCF COUPLE,TYPE=LOGR,PCOUPLE=(primarydataset,volser),
ACOUPLE=(alternatedataset,volser)
```

If there is a problem with the existing logger couple data sets(s),

1. Issue the following command to change to the logger alternate data set.

```
SETXCF COUPLE,TYPE=LOGR,ACOUPLE=(alternatedataset,volser)
```

2. Issue the following command to switch the current alternate logger couple data set to be the new primary couple data set.

```
SETXCF COUPLE,TYPE=LOGR,PSWITCH
```

If you are unable to resolve the problem at this point, it may be necessary to stop and restart logger. Before doing this, obtain a dump of logger on the systems that are impacted. See [z/OS MVS Diagnosis: Tools and Service Aids](#) for details on obtaining a logger dump.

See [z/OS MVS Diagnosis: Reference](#) for additional information on stopping and restarting the IXGLOGR address space.

To terminate logger, issue the following command. Note that this will cause all active connections on the system to be terminated. Message IXG056I will be issued when the IXGLOGR address space has terminated.

```
FORCE IXGLOGR,ARM
```

It may also be necessary to terminate structure connections by issuing the following command. The system will no longer have any type of connection to the structure (*strname*). If no other system has a connection to the structure, XES will deallocate it.

```
SETXCF FORCE,CON,STRNAME=strname,CONNAME=ALL
```

Once logger has terminated, it can be restarted by issuing the command S IXGLOGRS. After doing this, it may be necessary to restart applications that were previously connected. Logger will issue an ENF signal to indicate when it is ready to start work.

When the system logger address space fails

This topic applies to both coupling facility and DASD-only log streams, with differences noted in the text.

If the system logger address space fails, the following processing occurs:

- Log stream connections are terminated.
- The system issues a X'1C5' abend code.
- Any system logger requests from the system where the system logger component failed are rejected.

Contact the IBM Support Center for a fix for the problem, providing the SVC dump that accompanies abend code X'1C5'.

Subsystems or applications that held the lost connections may be required to be restarted. To start the system logger address space again after a failure, use the IXGLOGRS procedure in SYS1.PROCLIB:

```
START IXGLOGRS
```

This procedure starts the system logger address space as a system address space. At z/OS V1R5 or higher, or on a system with OW53349 applied, system logger will be automatically restarted after a failure. Check for messages IXG056I and IXG067E, which may be issued during the restart.

For coupling facility log streams, recovery processing for the system logger address space is done by peer connectors, if available, and consists of either offloading the coupling facility log data for the system where the failure occurred to DASD log data sets so that it is on a persistent media or ensuring that the log data is secure in a persistent duplex-mode list structure. The recovery process for the system logger address spaces is the same as that for system failures. See [“When an MVS system fails” on page 301](#) for a description of the recovery process.

When the coupling facility structure fails

This topic applies to coupling facility log streams only.

The following coupling facility problems can occur, resulting in rebuild processing for the structure:

- Damage to or failure of the coupling facility structure.
- Loss of connectivity to a coupling facility.
- A coupling facility becomes volatile.

For complete rebuild processing, see [z/OS MVS Programming: Sysplex Services Guide](#).

Damage to or failure of the coupling facility structure

If a structure was in duplex-mode and a coupling facility failure or structure damage occurs to only one instance of the structure, then XES will automatically switch from duplex-mode to simplex-mode. Logger will only be notified of the mode switch change and will not be aware of any coupling facility failure or damage to the structure. See [“When the coupling facility structure duplex/simplex mode changes” on page 308](#) for how Logger handles mode switches for the structure.

If the structure was in simplex-mode and the failure or damage occurs to the only instance of the structure, then all systems connected to the coupling facility structure detect the failure. The first system whose system logger component detects the failure initiates the structure rebuild process. The structure rebuild process results in the recovery of one or more of the affected coupling facility structure's log streams. All the systems in the sysplex that are connected to the list structure participate in the process of rebuilding the log streams in a new coupling facility list structure.

While the rebuild is in progress, system logger rejects any system logger service requests against the log stream. The status will be one of the following:

- The structure rebuild has completed successfully, the coupling facility structure and associated log streams are available, and system logger requests will be accepted.
- The structure rebuild was unsuccessful and connection to the structure is not possible because the structure is in a failed state. Log data still resides in staging data sets if they are used to duplex the log data for the log stream. If staging data sets were not used, the data persists in the local storage buffers on each system.

Loss of connectivity to the coupling facility structure

If a structure was in duplex-mode and a loss of connectivity occurs to only one instance of the structure (due to a hardware link failure), then XES will automatically switch from duplex-mode to simplex-mode. Logger will only be notified of the mode switch change and will not be aware of any loss of connectivity to the structure. See [“When the coupling facility structure duplex/simplex mode changes” on page 308](#) for how Logger handles mode switches for the structure.

For a simplex-mode structure, Logger detects the loss of connectivity to the single instance of the structure. Then, based on the rebuild threshold specified, if any, in the structure definition in the CFRM policy, the system that lost connectivity may initiate a rebuild for the structure.

If a rebuild is initiated, the event 48 parameter list mapped by macro IXGENF has bits IXGENFLogStreamsNotAvailable, and IXGENFStrRebuildStart, on, and field IXGENFStrName contains the name of the coupling facility structure affected. System logger rejects logger service requests issued during the rebuild process.

If XES cannot allocate a new structure instance in a coupling facility that can be connected to all the affected systems, system logger does one of the following, depending on whether the system or systems that cannot connect to the new coupling facility structure were using staging data sets:

- If the system was using staging data sets, the rebuild process continues and the coupling facility log data for the system is recovered from the staging data sets.
- If the system was **not** using staging data sets, the rebuild process is stopped. The systems go back to using the source structure.

The systems that do not have connectivity to the old coupling facility structure issue an ENF 48 event indicating that they do not have connectivity to the log stream.

The systems that can connect to the source structure issue an ENF 48 event indicating that the log stream is available to that system and can resume use of the log stream.

The installation should either update the CFRM active policy to make the new coupling facility structure available to all the systems or else fix the hardware link problem and then have the operator initiate a rebuild for the structure so that all the original systems will have connectivity.

When the coupling facility structure volatility state changes

This topic applies to coupling facility log streams only. The following coupling facility volatility state changes can occur, which may result in different reactions by system logger.

- A coupling facility becomes volatile, or for a duplex-mode structure when the coupling facilities containing both structure instances become volatile.
- A coupling facility becomes non-volatile, or for a duplex-mode structure when either coupling facility containing the structure instances becomes non-volatile.

Logger's behavior to the state change depends on:

- The type of state change
- The current duplexing environment for the log streams in the affected structures, and
- The log stream and structure duplexing specifications from the LOGR couple data set and the CFRM policy.

Some of the environments may result in rebuild processing for the structure. For complete information about rebuild processing, see [z/OS MVS Programming: Sysplex Services Guide](#).

Other environments may result in logger changing its duplexing method. Additionally, some of the environments may require no recovery nor duplexing changes by logger. See [“Selecting a method of duplexing coupling facility log data”](#) on page 283 for additional details on log data duplexing specifications.

A coupling facility becomes volatile

If a coupling facility changes to the volatile state, the system logger on each system using the coupling facility structure is notified. For structures that are in simplex-mode, a dynamic rebuild of the structure is initiated so that the log data can be moved to a non-volatile coupling facility. During rebuild processing, system logger rejects any logger service requests. If there is not a structure available in a non-volatile coupling facility, system logger will still rebuild the data on a new volatile coupling facility.

System logger may then change the way it duplexes coupling facility data because the volatile coupling facility constitutes a single point of failure:

- For log streams defined with STG_DUPLEX=YES, system logger will begin duplexing data to staging data sets, if they were not already in use.
- For log streams defined with STG_DUPLEX=NO, system logger will keep on duplexing data to local storage buffers on each system.

For duplex-mode structures, dynamic structure rebuild is not supported because there are already two copies of the structure. However, system logger may change whether it will duplex the log data and the way it duplexes the data since the new composite coupling facility volatility state constitutes a single point of failure.

- Assume the initial environment had a non-volatile composite structure coupling facility state but there was a failure-dependent relationship between the connecting system and the composite structure view.

The log stream was considered to be in a single point of failure environment and still is after the state change, so logger makes no changes to the duplexed method.

- Assume the initial environment had a non-volatile composite structure coupling facility state and there was a failure-independent connection between the connecting system and the composite structure view, but there was a failure-dependent relationship between the two structure instances.

The log stream was not considered to be in a single point of failure environment, but would be after the coupling facility state changed to volatile. Logger would continue to duplex the log data, but might change the duplexing method.

- For log streams defined with `STG_DUPLEX=YES`, system logger will begin duplexing data to staging data sets, if they were not already in use.

- Assume the initial environment had a non-volatile composite structure coupling facility state, there was a failure-independent connection between the connecting system and the composite structure view, and there was a failure-independent relationship between the two structure instances.

The log stream was not considered to be in a single point of failure environment, but would be after the coupling facility state changed to volatile.

- For log streams defined with `LOGGERDUPLEX(UNCOND)`, system logger will continue to duplex the log data. However, for log streams defined with `STG_DUPLEX=YES`, system logger will begin duplexing data to staging data sets, if they were not already in use.

- For log streams defined with `LOGGERDUPLEX(COND)`, system logger will first offload the log data from the structure, then begin duplexing any new log data written to the log stream.

- For log streams defined with `STG_DUPLEX=NO`, logger will begin duplexing data to local buffers.
- For log streams defined with `STG_DUPLEX=YES`, logger will begin duplexing data to staging data sets.

A coupling facility becomes non-volatile

If a coupling facility changes to a non-volatile state, the system logger on each system using the coupling facility structure is notified.

For simplex-mode structures, system logger may change the way it duplexes coupling facility data because the change to a non-volatile coupling facility may have removed the single point of failure condition.

- Assume the initial environment had a volatile structure coupling facility state and there was a failure-dependent relationship between the connecting system and the structure.

The log stream was considered to be in a single point of failure environment and continues to be after the state change, so system logger makes no changes to the duplexing method.

- Assume the initial environment had a volatile structure coupling facility state and there was a failure-independent relationship between the connecting system and the structure.

System logger may then change the way it duplexes coupling facility data because the non-volatile coupling facility no longer constitutes a single point of failure.

- System logger will duplex the log data using local buffers, unless the log stream definition specified `STG_DUPLEX(YES) DUPLEXMODE(UNCOND)`.

For duplex-mode structures, system logger may also change the way it duplexes coupling facility data because the change to a non-volatile coupling facility may have removed the single point of failure condition.

- Assume the initial environment had a volatile composite structure coupling facility state but there was a failure-dependent relationship between the connecting system and the composite structure view.

The log stream was considered to be in a single point of failure environment and continues to be after the state change (because of the failure-dependent connection), so system logger makes no changes to the duplexing method.

- Assume the initial environment had a volatile composite structure coupling facility state and there was a failure-independent connection between the connecting system and the composite structure view, but there was a failure-dependent relationship between the two structure instances.

The log stream was considered to be in a single point of failure environment, but would not be after the coupling facility state changes to non-volatile. System logger would continue to duplex the log data, but might change the duplexing method.

- The data will be duplexed by system logger using local buffers, unless staging data sets were specifically requested by having STG_DUPLEX(YES) DUPLEXMODE(UNCOND) on the log stream definition.
- Assume the initial environment had a volatile composite structure coupling facility state, there was a failure-independent connection between the connecting system and the composite structure view, and there was a failure-independent relationship between the two structure instances.

The log stream was considered to be in a single point of failure environment, but would not be after the coupling facility state changed to non-volatile.

- For log streams defined with LOGGERDUPLEX(UNCOND), system logger will continue to duplex the log data.
 - The data will be duplexed by system logger using local buffers, unless staging data sets were specifically requested by having STG_DUPLEX(YES) DUPLEXMODE(UNCOND) on the log stream definition.
- For log streams defined with LOGGERDUPLEX(COND), system logger will stop providing its own duplexing because the system-managed duplexed structure provides sufficient back-up capability for the log data.

When the coupling facility structure duplex/simplex mode changes

This topic applies to coupling facility log streams only. If a coupling facility structure changes modes, the system logger on each system using the coupling facility structure is notified.

System logger's behavior after a structure switches from simplex-mode to duplex-mode or vice versa depends on whether a single point of failure environment exists after the mode switch and the duplexing specifications for the structure and the log streams within the structure.

It is possible for system logger to change from using local buffers to staging data sets or to change from using staging data sets to local buffers. It is also possible for logger to start providing its own duplexing of the log data or stop duplexing the log data after the mode switch.

See “[Selecting a method of duplexing coupling facility log data](#)” on page 283 for additional details about log data duplexing specifications and considerations. In particular, see [Table 23 on page 287](#).

When the coupling facility space for a log stream becomes full

This topic applies to coupling facility log streams only.

Ordinarily, system logger off-loads coupling facility resident log stream data to DASD log data sets before the coupling facility storage allocated to the log stream is fully utilized. Occasionally however, the coupling facility storage allocated to a log stream reaches 100% utilization, for reasons such as a sudden burst of logging activity or because your coupling facility is sized too small for the volume of log data.

You can monitor the number of times a log stream becomes full using SMF record 88 (field SMF88ESF).

If you want to enlarge your coupling facility structure, you can do it in one of two ways:

- Alter the coupling facility size dynamically by issuing the SETXCF START,ALTER command or the IXLALTER service.
- Update the CFRM policy with the new coupling facility size, activate the policy, and then have the operator initiate a rebuild of the structure.

When a staging data set becomes full

This topic applies to both coupling facility and DASD-only log streams.

The staging data sets for each system should not fill up. If they do, you probably have them sized too small for your volume of log data and should enlarge them.

System logger will immediately begin offloading log data to DASD log data sets. If your staging data set is too small, you might find that offloading occurs very frequently.

You can monitor staging data set usage using SMF record 88, field SMF88ETF, for staging data set full conditions.

When a log stream is damaged

This topic applies to both coupling facility and DASD-only log streams.

System logger marks a log stream as permanently damaged when it cannot recover log data from either DASD staging data sets or the local storage buffers after a system, sysplex, or coupling facility failure. Applications are notified of the damage via system logger services. Recovery actions are necessary only if warranted for the application.

When DASD log data set space fills

This step applies to both coupling facility and DASD-only log streams.

The number of DASD log data sets available for log streams in a sysplex depends on whether you use the default (168 per log stream) or have provided additional directory extents in the LOGR couple data set. See [“Increasing the space available for DASD log data sets” on page 262](#) for more information.

System logger monitors usage of the available log stream directory space, notifying you as follows if you start to run out of space:

- If you are using the DSEXTENT parameter in the LOGR couple data set system logger issues messages IXG261E and IXG262A indicating that usage of directory extents is over 85% and 95% respectively.
- If you are using the default number of log data sets allowed for a log stream (168), system logger issues message IXG257I indicating that the data set directory for the log stream is over 90% full.

If you have run out of log stream directory space, off-loads may fail. When this occurs, system logger issues message IXG301I. Offload processing for the log stream cannot complete until more log stream directory space or directory extents are made available. If the last disconnect to a log stream occurs and the offload cannot complete successfully, the log stream is in a failed state. In this case, the log stream is considered 'in use' and there may be a failed-persistent connection to a structure associated with the log stream.

You can make more directory space available for a log stream in one of the ways below. Use the IXCMIAPU utility to run a report of the log stream definitions in the LOGR couple data set to help you with this step. The LIST LOGSTREAM NAME(*) DETAIL(YES) statement outputs information showing which log streams might be using large numbers of data sets and directory extents. See [“LOGR keywords and parameters for the administrative data utility” on page 358](#) for more information about IXCMIAPU.

- Format another set of LOGR couple data sets with a higher DSEXTENT value and bringing them into the sysplex as the active primary and alternate LOGR couple data sets. See [“Increasing the space available for DASD log data sets” on page 262](#). See Chapter 11, [“Format utility for couple data sets,” on page 313](#) for the IXCL1DSU utility and the DSEXTENT parameter.
- Free directory extents currently in use in one of the following ways:
 - Use a program that issues the IXGDELET service to delete enough data from the log stream to free up space in the log stream data set directory.

Some products provide a program to delete log stream data. See [“Deleting log data and log data sets” on page 291](#) for deletion programs provided for IBM products.

- Delete log stream definitions from the LOGR couple data set.

Identify and delete the definitions for unused or unnecessary log streams. This will free the directory space associated with the log streams, which may free up directory extents for use by other log streams.

Note: Deleting DASD log data sets using a non-system logger method will not work because system logger will still count the data sets toward the data set directory entry limit. You cannot, for example:

- Use a TSO/E DELETE command to delete a log data set.
- Use DFHSM to migrate log stream data sets to tape.

When unrecoverable DASD I/O errors occur

This topic applies to both coupling facility and DASD-only log streams, with differences noted.

DASD I/O errors may occur against either log data sets or staging data sets. System logger tries to recover from the error, but if it cannot, the error is characterized as an **unrecoverable I/O error**. See the following:

- [“When unrecoverable DASD I/O errors occur during offload” on page 310](#)
- [“When staging data set unrecoverable DASD I/O errors occur” on page 310](#)

When unrecoverable DASD I/O errors occur during offload

DASD I/O errors may occur during offload processing, while log data is being written to DASD log data sets. When this happens, system logger tries to recover by closing the current log data set and allocating a new one. If this process fails, the I/O error is characterized as an unrecoverable I/O error.

In the case of unrecoverable I/O errors, system logger will accept subsequent IXGWRITE requests as follows:

- For a coupling facility log stream, system logger will accept IXGWRITE requests if the log stream is connected to a coupling facility where there is still room for log data. If the coupling facility is full or no coupling facility exists, system logger rejects IXGWRITE requests.
- For a DASD-only log stream, system logger will accept IXGWRITE requests until the staging data set for the system writing to the log stream is filled.

IXGBRWSE and IXGDELET requests may continue to work. I/O errors encountered in the process of completing these requests are reported to the application in return and reason codes.

To correct an unrecoverable I/O problem, delete the log stream definition in the LOGR policy and redefine it with different log data set attributes, such as LS_DATACLAS, in order to get the log stream data set allocated in a usable location.

When staging data set unrecoverable DASD I/O errors occur

DASD I/O errors may occur when log data is being duplexed to DASD staging data sets. When this occurs, system logger tries to recover by doing the following:

1. Offload current log data to DASD log data sets.
2. Delete and unallocate the staging data set.
3. Re-allocate a new instance of the staging data set.

In the meantime, system logger continues to accept write and other requests against the log stream.

If system logger cannot re-allocate a new staging data set, the I/O error is characterized as unrecoverable. In the case of an unrecoverable staging data set I/O error, system logger does the following:

- **For a coupling facility based log stream**, system logger switches the duplexing mode to duplex log data to local storage buffers. The system issues message IXG255I indicating that the duplexing mode has changed. Normal system logger processing continues. The log stream may be more vulnerable to data loss due to system, sysplex, or coupling facility failure.
- **For a DASD-only log stream**, system logger disconnects connectors from the log stream. The system issues message IXG216I to indicate that connectors are being disconnected because a staging data set could not be allocated.

Preparing for z/OS IBM zAware log stream client usage

z/OS provides the capability of having specific log stream data sent "out-of-band" to the IBM Z Advanced Workload Analysis Reporter (IBM zAware) firmware. IBM zAware is described in the IBM Z Advanced Workload Analysis Reporter (IBM zAware) Guide.

From a z/OS perspective, the z/OS images communicating with an IBM zAware firmware are identified as z/OS IBM zAware clients, and the IBM zAware firmware is identified as the IBM zAware server. That server will monitor/manage the z/OS images within its scope.

For z/OS, the initial data being sent to the IBM zAware server for analysis is the log data within the Operlog log stream, which is called a z/OS IBM zAware log stream client. This allows the IBM zAware server to provide analytical monitoring and machine learning of z/OS health for purposes of availability management.

By default, z/OS will not communicate with the IBM zAware server unless the z/OS IBM zAware log stream client requirements are established as follows:

1. Specify the ZAI parameters for the z/OS IBM zAware log stream client socket communication and log data buffering details in order to send log stream data to the IBM zAware server that will receive and process data from this z/OS image. See the system logger SYS1.PARMLIB member IXGCNFxx in [z/OS MVS Initialization and Tuning Reference](#).
2. The IXGLOGR address space must have security permission for the OMVS segment. The OMVS segment is only for TCP/IP connectivity. UID(0) or superuser ability can be used but are not required. For example, in RACF issue the following command(s), where xxxx is a unique user ID and yyyy is a unique group ID:

```
ADDUSER IXGLOGR OMVS(UID(xxxx) HOME('/'))
```

or

```
ADDGROUP IXGGRP OMVS(GID/yyyy)
ADDUSER IXGLOGR DFLTGRP(IXGGRP) OMVS(UID(xxxx) HOME('/tmp'))
PROGRAM('/bin/false') NOPASSWORD
```

3. The z/OS Communications Server environment must be available, that is, the z/OS UNIX System Services (OMVS) and resolver address spaces, VTAM address space and appropriate TCP/IP address space have been started. Also, the necessary TCP/IP (network) definitions provided for the server location to be determined in order for logger to establish a (socket) connection to the IBM zAware server. See [z/OS Communications Server: IP Configuration Guide](#) and [z/OS UNIX System Services Planning](#) for additional details for establishing the desired environment.
4. The appropriate log streams have the ZAI-related keywords specified for the intended log stream data to be sent to the IBM zAware server. For example, use the ZAI(YES) ZAIDATA('OPERLOG') parameter specifications for the SYSPLEX OPERLOG log stream for OPERLOG log data to be sent to the IBM zAware server.

If the System Authorization Facility (SAF) is available, SAF update access to the RESOURCE (IXGZAWARE_CLIENT) in class CLASS(FACILITY) is required for the ZAI(YES) log stream specification. Also, the active primary TYPE=LOGR couple data set in the sysplex must be formatted at an HBB7705 level or higher in order to specify the keywords.

It is important to note that any log streams defined using the LIKE(SYSPLEX.OPERLOG) specification will obtain the ZAI and ZAIDATA setting from the SYSPLEX.OPERLOG log stream definition. The installation needs to take care to ensure that these other log streams do not have the ZAI(YES) designation, unless that is the absolute intention.

5. When a log stream connection exists on the z/OS image for the appropriate log stream, and the above conditions have been met, system logger will establish z/OS z/OS IBM zAware log stream client (socket) connection and send the log data to the appropriate IBM zAware server.

System logger uses "IBM ZAI LOGSTREAM CLIENT" in message text to see "z/OS IBM zAware log stream client" activity.

If the IBM zAware server is not available for a short period (that is if it has been deactivated for maintenance), the BPX-services that system logger uses to send data will fail with error return codes. The system logger will discover that either the socket connection is lost or after a number of retries on sending data that continues to fail, system logger will disconnect from the socket explicitly. System logger will then attempt a new socket connection with the IBM zAware server. If the connection succeeds, system logger will continue sending written OPERLOG messages to the IBM zAware server.

If the second attempt to connect to the server does not succeed, then the z/OS IBM zAware log stream client is considered in a quiesced state. This means that any OPERLOG messages that were buffered for the purpose of sending to the IBM zAware server will be tossed and any newly written OPERLOG messages will not be buffered. After the IBM zAware server is made available again, z/OS system logger provides a SETLOGR FORCE,ZAICONN,LSN=SYSPLEX.OPERLOG command to cause system logger to establish a socket connection with that server. If this connection succeeds, system logger will start sending newly written OPERLOG message to the IBM zAware server.

If the IBM zAware server is not available for a short period (for example, if it has been deactivated for maintenance), the BPX-service that system logger uses to send data will fail with error return codes. The system logger will discover that either the socket connection is lost or after a number of retries on sending data that continue to fail, system logger will disconnect from the socket explicitly. At this point, the z/OS IBM zAware log stream client is considered in a quiesced state. This means that any operlog messages that were buffered for the purpose of sending to the IBM zAware server will be tossed and any newly written operlog message will not be buffered.

After the IBM zAware server is made available again, z/OS system logger will provide for a SETLOGR FORCE,ZAICONN,LSN=SYSPLEX.OPERLOG command to cause system logger to establish a socket connection with the IBM zAware server. If the connection and hand shaking succeeds, system logger will resume sending newly written operlog messages to the IBM zAware server.

Chapter 11. Format utility for couple data sets

IBM provides the couple data set format utility to allocate and format the DASD couple data sets for your sysplex. See [Chapter 3, “Planning the couple data sets,”](#) on page 27 for couple data set considerations before proceeding with couple data set formatting. **In particular, note the recommendations about not over-specifying the size or the parameter values in a policy, which could cause degraded performance in a sysplex.**

The control statements for the utility programs follow standard conventions for JCL statements. The utility program accepts 80-byte records that can contain one or more parameters per record. Columns 72 through 80 of the record image are not used. See [z/OS MVS JCL Reference](#) for additional information about JCL statements.

Use the XCF couple data set format utility to create and format all sysplex couple data sets prior to IPLing a system that is to use the sysplex couple data sets. Other types of couple data sets do not have to be formatted prior to IPLing the system. The format utility is portable; that is, you can run the utility on any z/OS system.

Using the couple data set format utility

The name of the XCF couple data set format utility is IXCL1DSU. This program resides in SYS1.MIGLIB (which is logically appended to the LINKLIST). The utility is available through STEPLIB, which allows you to run older versions of the utility if you want (for example, from previous z/OS releases).

IXCL1DSU in z/OS allows you to format all types of couple data sets for your sysplex. The utility contains two levels of format control statements. The primary format control statement, DEFINEDS, identifies the couple data set being formatted. The secondary format control statement, DATA TYPE, identifies the type of data to be supported in the couple data set — sysplex data, automatic restart management data, CFRM data, SFM data, WLM data, LOGR data, LOGRY data, LOGRZ data, or z/OS UNIX System Services data.

- Once the ARM, CFRM, SFM, LOGR, LOGRY or LOGRZ couple data sets are formatted, you use another utility program, IXCMIAPU, to create or update administrative data for these services. See [Chapter 12, “Administrative data utility,”](#) on page 335 for a description of the IXCMIAPU utility program.
- For WLM couple data sets, you use an ISPF interface to create policy administrative data. See [z/OS MVS Planning: Workload Management](#) for a description of the ISPF interface.
- For UNIX System Services couple data sets, you do not need to perform any additional steps.

Note that sysplex couple data sets need no additional processing.

The sysplex couple data set must be formatted to contain only XCF control data. For other types of couple data sets, however, you have the option of either formatting a couple data set for each type of service separately or combining more than one type of service in a single couple data set. Thus, you could format a CFRM couple data set and an SFM couple data set, or you could combine the utility control statements to format a CFRM/SFM couple data set. The system maintains the data for each type of service separately so that data integrity is ensured.

If you choose to format separate couple data sets for the services you are using, you can place them on the same DASD volume or on separate volumes, depending on your capacity and performance requirements.

Understanding couple data set coexistence

The IXCL1DSU format utility allows you to specify different maximum values for some couple data set parameters. These values are based on the version of the code installed on your system. For example, the initial maximum value for the CFRM parameters STR and CONNECT was 64. Subsequent updated versions of the utility allowed you to specify 255 as the maximum value for CONNECT and 1024 as the maximum

value for STR. These increases cause a corresponding increase in the physical size of the CFRM couple data set and the individual records the data set contains.

In a sysplex where not all systems support the same couple data set types, or maximum values for particular couple data set types, special attention must be given to uplevel and downlevel data sets. (In this context, an uplevel couple data set refers to one that has been formatted to:

- Support a new item type that did not exist in a prior version of the couple data set.
- Support more instances of a particular type of item than were supported in a prior version of the couple data set.
- Support both examples cited above.

A downlevel couple data set is one that has not been formatted as described for an uplevel couple data set.

The following coexistence concerns should be noted for uplevel and downlevel couple data sets:

- XCF does not allow you to install an uplevel couple data set on a system in a sysplex in which one or more systems do **not** include the code for a newer version of the format utility (and for the higher maximum values specified in the utility).

For example, you cannot issue a SETXCF COUPLE command to bring into use a CFRM couple data set formatted for 255 structures in a sysplex in which the code support for the 255 maximum has not been installed on all systems.

- When defining your primary and alternate couple data sets, the system does not allow you to mix couple data sets formatted at different versions when running in a sysplex in which one or more systems do **not** include the code for a newer version of the format utility (and for the higher maximum values specified in the utility).

For example, do not specify a primary couple data set formatted for 64 structures and an alternate couple data set formatted for 255 structures on a system on which the code support for the 255 maximum has not been installed.

Coding the couple data set format utility

See the following figures for examples of how to set up your JCL to format different types of couple data sets:

- For a sysplex couple data set, see [“Formatting a sysplex couple data set” on page 331](#)
- For an ARM couple data set, see [“Formatting an automatic restart management couple data set” on page 331](#)
- For a CFRM couple data set, see [“Formatting a CFRM couple data set” on page 331](#)
- For a LOGR couple data set, see [“Formatting a sysplex scope LOGR couple data set” on page 327](#)
- For LOGRY and LOGRZ couple data sets, see [“Formatting a single-system scope LOGRY or LOGRZ couple data set” on page 328](#)
- For an SFM couple data set, see [“Formatting an SFM couple data set” on page 332](#)
- For a z/OS UNIX couple data set, see [“Formatting a UNIX System Services couple data set” on page 332](#)
- For a WLM couple data set, see [“Formatting a WLM couple data set” on page 332](#).

The utility control statements are described below:

PGM=IXCL1DSU

The format utility program that is shipped with z/OS.

STEPLIB

Identifies the SYS1.MIGLIB library.

SYSPRINT

Describes where the output messages from the format utility are to be written. The SYSPRINT DD statement is required.

SYSIN

Describes the utility control statements used to format one or more couple data sets.

DEFINEDS

Specifies the beginning of the definition of a new couple data set. There is no limit to the number of DEFINEDS statements that you provide, and therefore to the number of couple data sets that you format. The couple data set to be created should not have previously existed and must not be found by the format utility either on the specified volume or in the catalog. This prevents the format program from being run against a couple data set that is in use.

Specific Couple Data Set Information: The information for the **type** of couple data set being formatted is identified with parameters of the DEFINEDS statement. For a description of the keywords to format a couple data set type, see the respective topic:

- [“Sysplex parameters for format utility” on page 317](#)
- [“Automatic restart management parameters for format utility” on page 318](#)
- [“CFRM parameters for format utility” on page 319](#)
- [“LOGR parameters for format utility \(sysplex scope\)” on page 320](#)
- [“LOGRY or LOGRZ single-system scope parameters for format utility” on page 326](#)
- [“SFM parameters for format utility” on page 328](#)
- [“WLM parameters for format utility” on page 329](#)
- [“z/OS UNIX parameters for format utility” on page 328](#)

The physical attributes of the couple data set are controlled with the following parameters of DEFINEDS.

SYSPLEX(sysplex-name)

Specifies the 1 - 8 character name of the sysplex in which the couple data set is to be used. The valid characters are alphanumeric characters (A-Z and 0-9) and national characters (\$,@,#). The first character of the name must be alphabetic. The sysplex name is a required parameter.

MAXGROUP(total-group) THIS STATEMENT CAN BE USED ONLY ON A DEFINEDS STATEMENT WHEN FORMATTING A SYSPLEX COUPLE DATA SET. IT IS MAINTAINED FOR COMPATIBILITY WITH MVS/ESA SP VERSION 4 STATEMENTS.

IBM recommends use of DATA TYPE(SYSplex) with ITEM NAME(GROUP) instead of the MAXGROUP keyword. Specifies the number of groups to be supported by the new sysplex couple data set. If this parameter is omitted, a default of 10 is used. The maximum allowable value for this parameter is release dependent. In z/OS, the maximum supported is 2045. The minimum value of MAXGROUP allowed is 10.

Note that if DATA TYPE(SYSplex) is specified, the maximum number of groups allowed is expected to be specified using ITEM NAME(GROUP).

If MAXGROUP and DATA TYPE(SYSplex) are both specified:

- If ITEM NAME(GROUP) is specified, message IXC291I will be issued stating that the MAXGROUP value was overridden by the ITEM NAME(GROUP) value.
- If ITEM NAME(GROUP) is omitted, message IXC291I will be issued stating that MAXGROUP is incompatible with DATA TYPE(SYSplex) and the data set will not be formatted.

MAXMEMBER(members-per-group) THIS STATEMENT CAN BE USED ONLY ON A DEFINEDS STATEMENT WHEN FORMATTING A SYSplex COUPLE DATA SET. IT IS MAINTAINED FOR COMPATIBILITY WITH MVS/ESA SP VERSION 4 STATEMENTS.

IBM recommends use of DATA TYPE(SYSplex) with ITEM NAME(MEMBER) instead of the MAXMEMBER keyword. Specifies the maximum number of members in any one XCF GROUP. If this parameter is omitted, a default value is taken. The default number of members in a group is calculated to be the number of members which can be accommodated on one track on the device type on which the sysplex couple data set resides. The maximum allowable value for this parameter is release dependent. In z/OS, the maximum supported is 1023. The minimum value of MAXMEMBER allowed is 8. Systems joining the sysplex obtain the MAXMEMBER value from the

sysplex couple data set. The actual number defined by the format utility might be larger than the number specified. This is due to the fact that XCF adds one member for its own use and rounds the number of members to the next larger multiple of four.

Note that if DATA TYPE(SYSPLEX) is specified, the maximum number of members allowed is expected to be specified using ITEM NAME(MEMBER).

If MAXMEMBER and DATA TYPE(SYSPLEX) are both specified:

- If ITEM NAME(MEMBER) is specified, message IXC291I will be issued stating that the MAXMEMBER value was overridden by the ITEM NAME(MEMBER) value.
- If ITEM NAME(MEMBER) is omitted, message IXC291I will be issued stating that MAXMEMBER is incompatible with DATA TYPE(SYSPLEX) and the data set will not be formatted.

MAXSYSTEM(maximum-systems)

Specifies the number of systems to be supported by the new couple data set. If this parameter is omitted, the format utility assigns a default value of eight. In z/OS, the maximum value is 32.

When formatting the couple data set to contain the CFRM policy data, ensure that the value you specify for MAXSYSTEM matches the value for MAXSYSTEM that was used when the sysplex couple data set was formatted.

When coupling facility structures are used for XCF signaling and the MAXSYSTEM value specified for the CFRM couple data set is less than that of the sysplex couple data set, systems might not be able to join the sysplex. For example, if the only signaling connectivity is through coupling facility structures, MAXSYSTEM=16 is specified for the sysplex couple data set, and MAXSYSTEM=8 is specified for the CFRM couple data set, then at most eight systems will be allowed in the sysplex. (The limit might be less than eight as described in [“Implications of the MAXSYSTEM value for the sysplex couple data set”](#) on page 36.)

Note:

1. If MAXSYSTEM is greater than eight, and RMAX (the maximum length of WTOR reply IDs, as specified in the CONSOLxx parmlib member) is less than 99, then the system will increase RMAX to 99 when you start using this data set. This is to handle the potential increase in WTOR traffic in a sysplex of greater than eight systems. Once RMAX is increased, it cannot be decreased. The system issues message IEA403I to indicate that the maximum number of reply IDs becomes 99.
2. When a sysplex is configured with a MAXSYSTEM value greater than 8, reply IDs are no longer assigned in strict sequential order. Instead, systems obtain groups of reply IDs for assignment to WTORs, and the IDs might not be assigned in sequential order. This change requires no coding changes on the installation's part, but might surprise an operator. You should consider informing operators of this change.

DSN(data-set-name)

Specifies the 1 - 44 character length name to be used for the couple data set. The format utility will allocate a new data set by this name on the volume specified by the VOLSER parameter. The valid characters are alphanumeric characters (A-Z and 0-9), national characters (\$,@,#), a period (.), and a hyphen (-). The first character must be either an alphabetic or a national character. DSN is a required parameter.

VOLSER(volser)

Specifies the 1 - 6 character length name of the volume on which the couple data set is to be allocated. The valid characters are alphanumeric characters (A-Z and 0-9) and national characters (\$,@,#). VOLSER is a required parameter for non-SMS managed data sets. For SMS-managed data sets and VOLSER restrictions, see the STORCLAS and MGMTCLAS parameters.

UNIT(type)

Specifies the type of the unit to which the couple data set is to be allocated. You can specify an installation-defined group name, generic device type, or specific device number. The UNIT parameter is optional. If you specify UNIT, then you must also specify VOLSER.

NOCATALOG**CATALOG**

Specifies whether the data set is to be cataloged. The default is not to catalog the data set if neither CATALOG nor NOCATALOG is specified. If the data set is SMS-managed, NOCATALOG is ignored.

STORCLAS(storage-class)

Specifies the SMS storage class for the SMS-managed volume where the data set is to reside. **storage-class** can be any valid SMS storage class name. STORCLAS is an optional parameter.

Either STORCLAS or VOLSER must be defined on the DEFINEDS statement. If STORCLAS is specified, you do not need to specify VOLSER. It is recommended that you specify the GUARANTEED SPACE=YES attribute for the storage class. You can use ISMF panels to specify names of storage classes with the guaranteed space attribute.

If your installation has assigned storage classes and management classes through automatic class selection (ACS) routines, the routines might affect where SMS places the couple data set. The format utility issues a message to indicate the volume that SMS has selected for the data set.

MGMTCLAS(management-class)

Specifies the SMS management class for the SMS-managed volume where the data set is to reside. **management-class** can be any valid SMS management class name. MGMTCLAS is an optional parameter.

If your installation has assigned storage classes and management classes through automatic class selection (ACS) routines, the routines might affect where SMS places the couple data set. The format utility issues a message to indicate the volume that SMS has selected for the data set.

DATA TYPE

Indicates the type of data to be supported by the couple data set. Use one DATA TYPE control statement to identify each function to be supported by the couple data set.

DATA TYPE(SYSPLEX)**DATA TYPE(ARM)****DATA TYPE(CFRM)****DATA TYPE(LOGR)****DATA TYPE(SFM)****DATA TYPE (WLM)****DATA TYPE(BPXMCDs)**

Identifies the type of data to be supported by the couple data set. Note that SYSPLEX data cannot be combined with other data types in the same couple data set.

ITEM NAME(data-name) NUMBER(data-number)

Specifies the number of occurrences of the data specified on the DATA TYPE statement that are to be supported on the couple data set.

Sysplex parameters for format utility

For a sysplex couple data set — DATA TYPE(SYSPLEX), valid data names are as follows:

ITEM NAME(GROUP) NUMBER()

Specifies that the sysplex couple data set supports group data. Include not only the number of XCF groups needed for multisystem applications, but also the number of XCF groups that MVS system components need. (To determine the number of MVS system groups, issue the DISPLAY XCF,G command.) It is also advisable to add a contingency number of groups for future growth. See [“Considerations for all couple data sets” on page 27](#). (Default=50, Minimum=10, Maximum=2045)

ITEM NAME(MEMBER) NUMBER()

Specifies that the sysplex couple data set supports member data. Determine which group has the largest number of members that can exist on systems in a sysplex and specify a value at least as large. (Default=number of members that can be supported on one track of the device type for which the sysplex couple data set is being formatted, Minimum=8, Maximum=2047.)

ITEM NAME(GRS) NUMBER(1)

Specifies that the sysplex couple data set supports global resource serialization data.

ITEM NAME(CLUSTER) NUMBER(1)

Specifies that the sysplex couple data set supports cluster functions. There must be at least one z/OS system at release level V1R9 or higher to enable the system to process this parameter. You will need to specify the ITEM NAME(CLUSTER) parameter in your primary sysplex couple data set in order to enable any of the operations that report or act on cluster resources.

ITEM NAME(SSTATDET) NUMBER(1)

Specifies that the sysplex couple data set is formatted with the larger system records required by the system status detection partitioning protocol.

Automatic restart management parameters for format utility

For an automatic restart management couple data set — DATA TYPE(ARM) — valid data names are POLICY, MAXELEM, and TOTELEM.

ITEM NAME(POLICY) NUMBER()

Specifies the number of user-defined administrative policies that can be defined. (Default=3, Minimum=1. Maximum=65535.)

ITEM NAME(MAXELEM) NUMBER()

Specifies the maximum number of elements per policy. (Default=10, Minimum=1. Maximum=65535.)

ITEM NAME(TOTELEM) NUMBER()

Specifies the maximum number of elements that can be registered in the sysplex at once. (Default=500, Minimum=1. Maximum=65535.)

ARM couple data set versioning — new format level

Automatic restart management allows different levels or functional versions of ARM couple data sets to be defined. The following couple data set format levels or functional versions can be created using the IXCL1DSU utility.

- Base format level. This is the initial or base ARM couple data set format level and is created when the ARM couple data set is formatted using a version of IXCL1DSU prior to z/OS V1R4.
- z/OS V1R4 format level (HBB7707 SYMBOL TABLE SUPPORT). This format level is created when the ARM couple data set is formatted using a version of IXCL1DSU at z/OS V1R4 or higher.
- z/OS V2R2 format level (HBB77A70 SYMBOL TABLE SUPPORT). This format level is created when the ARM couple data set is formatted using a version of IXCL1DSU at z/OS V2R2 or higher.

ARM couple data sets formatted with the z/OS V2R2 version of IXCL1DSU are required in order to support the larger symbolic substitution table that is available in V2R2. Systems at different system levels can access the V2R2-formatted ARM couple data set.

1. Using the z/OS V2R2 version of IXCL1DSU, define new primary and alternate ARM couple data sets.
2. Enable the new couple data set as the primary couple data set as follows:

If there is already a primary ARM couple data set enabled in the sysplex (by another active system in the sysplex), use the SETXCF COUPLE system command to enable the couple data set as the alternate couple data set and then use the SETXCF PSWITCH system command to switch the alternate couple data set to the primary couple data set.

```
SETXCF COUPLE,TYPE=ARM,PSWITCH
```

These commands should be issued from a system where the ARM couple data set is enabled as the primary.

Once the ARM couple data set is enabled as the primary couple data set in the sysplex, V2R2 systems can be IPLed into the sysplex.

CFRM parameters for format utility

When formatting the couple data set to support CFRM policy data, the value that is specified for MAXSYSTEM should match the value that is specified for MAXSYSTEM when formatting the sysplex couple data sets. For a CFRM couple data set — DATA TYPE(CFRM)—valid data names are POLICY, CF, STR, CONNECT, SMREBLD, SMDUPLEX, MSGBASED, ASYNCDUPLEX., and .

ITEM NAME(POLICY) NUMBER()

Specifies the number of administrative policies that can be defined within the data set. (Default=6, Minimum=1, Maximum=32)

ITEM NAME(CF) NUMBER()

Specifies the number of coupling facilities that can be defined within a policy. (Default=8, Minimum=1, Maximum=16)

ITEM NAME(STR) NUMBER()

Specifies the number of coupling facility structures that can be defined in one policy. This number refers to structure definitions, not to the number of allocated structure instances. One structure definition can refer to more than one allocated structure instance, as would be the case for a structure that was being rebuilt or duplexed. (Default = 50, Minimum = 1, Maximum = 2048)

ITEM NAME(CONNECT) NUMBER()

Specifies the number of connections that can be defined for a coupling facility structure. (Default = 32, Minimum = 1, Maximum = 255)

ITEM NAME(SMREBLD) NUMBER(1)

Specifies that this CFRM couple data set supports the system-managed rebuild process.

ITEM NAME(SMDUPLEX) NUMBER(1)

Specifies that this CFRM couple data set supports the system-managed duplexing rebuild process. Use this statement only when formatting a CFRM couple data set for systems at z/OS Release 2 or later. Specifying SMDUPLEX implies support for SMREBLD, regardless of whether SMREBLD was explicitly specified.

ITEM NAME(MSGBASED) NUMBER(1)

Specifies that this CFRM couple data set supports message-based event and confirmation processing. Use this statement only when formatting a CFRM couple data sets at systems at z/OS Release 8 and later. Specifying MSGBASED implies support for SMREBLD and SMDUPLEX, regardless of whether they were explicitly specified.

ITEM NAME(ASYNCDUPLEX) NUMBER(1)

Specifies that this CFRM couple data set supports system-managed asynchronous structure duplexing. Specifying ASYNCDUPLEX implies support for SMREBLD, SMDUPLEX, and MSGBASED, regardless of whether they were explicitly specified.

Considerations for supporting CFRM functions

Certain CFRM functions require specific levels of z/OS and restrict the use of CFRM couple data sets that were formatted for the use of the function. [Table 25 on page 319](#) lists the functions that have such requirements and restrictions, the keyword that is specified when formatting the CFRM couple data set, and the minimum release level that systems must be running in order to use the CFRM couple data set.

Table 25. Support for CFRM functions			
Function	Data	Minimum Release	Notes
System-managed asynchronous duplexing rebuild	ASYNCDUPLEX	z/OS V2R2 with APAR OA47796 installed	
> 1024 structure definitions	STR	z/OS V1R10	
Message-based processing	MSGBASED	z/OS V1R8	.

Table 25. Support for CFRM functions (continued)			
Function	Data	Minimum Release	Notes
> 512 structure definitions	STR	z/OS V1R4	
System-managed duplexing rebuild	SMDUPLEX	z/OS V1R2	
System-managed rebuild	SMREBLD	OS/390 V2R8	None.
> 255 structure definitions	STR		None.

Two main considerations apply to each of the functions that are listed in [Table 25 on page 319](#):

- Only systems at the listed minimum release level are capable of using a CFRM couple data set formatted for the particular function. For example, only systems at z/OS V1R8 are capable of using a CFRM couple data set formatted for message-based processing. Once a version of the CFRM couple data set that was formatted for the function (by specifying the Data Name keyword) is in use as the primary or alternate CFRM couple data set, any systems at a release level lower than the minimum that join the sysplex will be unable to access the CFRM couple data set. The inability to access the CFRM couple data set means that those systems cannot exploit any CFRM function, such as connecting to a coupling facility structure, submitting coupling facility requests, and issuing DISPLAY XCF,CF and DISPLAY XCF,STR commands.
- Once a version of a CFRM couple data set that is formatted to support a particular function is made the primary CFRM couple data set, the sysplex cannot revert to a CFRM couple data set that was formatted at a lower version without a sysplex-wide IPL. (However, it should not be necessary to fall back from a CFRM couple data set that supports a system-managed process to one that does not. For example, for system-managed duplexing rebuild, an alternative would be to disable duplexing for all structures by changing the policy to one that specifies or defaults to DUPLEX(DISABLED) for all structures, or some desired subset of structures.)

The sysplex will not permit a data set formatted for less capability to become the alternate CFRM couple data set. Use the D XCF,COUPLE,TYPE=CFRM command to determine the options specified to format the CFRM couple data set. The following format rules apply:

- The SETXCF command to identify an alternate couple data set

```
SETXCF COUPLE,ACOUPL=(alternatedsname,alternatevolume)
```

will be rejected if *alternatedsname* is not also a CFRM couple data set formatted to support at least the same level of system-managed process.

- The first system that IPLs into a sysplex cannot specify in COUPLExx a version of a primary CFRM couple data set that has been formatted to support a particular function and a version of an alternate CFRM couple data set that has not been formatted to support that function. The sysplex will reject the alternate and will operate without an alternate until a version of a CFRM couple data set that has been formatted to support at least the same level of function is made available as the alternate CFRM couple data set.

LOGR parameters for format utility (sysplex scope)

For a LOGR policy — DATA TYPE(LOGR), valid data names are LSR, LSTRR, DSEXTENT, and SMDUPLEX.

When formatting the couple data sets to support the system logger sysplex scope LOGR policy data, IBM recommends that the value specified for MAXSYSTEM should match the value specified for MAXSYSTEM when formatting the sysplex couple data sets.

ITEM NAME(LSR) NUMBER()

Specifies the maximum number of log streams that can be defined to the LOGR policy. Each LSR record consumes one track in the LOGR couple data set. (Default=1, Minimum=1, Maximum=32767)

ITEM NAME(LSTRR) NUMBER()

Specifies the maximum number of structure names that can be defined to the LOGR policy. Each LSTRR record consumes one track in the LOGR couple data set. (Default=1, Minimum=1, Maximum=32767)

ITEM NAME(DSEXTENT) NUMBER()

Specifies the number of additional log stream data set directory extents to define for log stream offload data sets. Use this parameter to increase the potential log data set directory capacity for a sysplex. Each additional directory extent specified goes into a common pool available to any log stream in the sysplex. System logger allocates these directory extents as needed when a log stream runs out of DASD log stream offload data set directory space. Each directory extent allows a log stream to extend by 168 log data sets.

When a DSEXTENT record is assigned to a log stream, that particular DSEXTENT record is only used for that specific log stream until it is no longer needed. Logger then removes the DSEXTENT record from that specific log stream use and put the DSEXTENT record back into the common pool.

Each DSEXTENT record consumes one track in the LOGR couple data set. (Default=0, Minimum=0, Maximum=65535)

See [“Increasing the space available for DASD log data sets” on page 262](#) for guidance on the DSEXTENT parameter.

ITEM NAME(SMDUPLEX) NUMBER()

Specifies whether Logger should support system-managed duplexing rebuild. This specification can also affect the LOGR couple data set format level. See [“LOGR couple data set versioning — new format levels” on page 321](#) for an explanation of the different versions or levels of LOGR couple data sets and the system logger functions that are supported. The value of the NUMBER parameter indicates one of the following:

Value**Description****0**

Logger does not support system-managed duplexing rebuild.

1

Logger does support system-managed duplexing rebuild. The resulting format level of the LOGR couple data set is HBB7705 and is supported on z/OS V1R2 and later releases.

This item can only be specified on systems at z/OS V1R2 and higher.

When this item is omitted from the IXCL1DSU format utility, logger handles it differently based on the z/OS release level. If this item is omitted on a z/OS V2R2 or higher release, then logger treats it as if a value of one (1) was specified. For earlier z/OS release levels, logger treats it as if a value of zero (0) was specified.

This specification does not consume any space in the LOGR couple data set. (Default=1, Minimum=0, Maximum=1)

LOGR couple data set versioning — new format levels

For general couple data set coexistence considerations, refer to [“Considerations for all couple data sets” on page 27](#) and [“Understanding couple data set coexistence” on page 313](#).

Logger allows different format levels or functional versions of LOGR couple data sets to be defined. Certain logger functions require a specific format level of the LOGR couple data set as well as a particular release level of z/OS. There are also minimum release levels required for all systems in a sysplex to use the different LOGR couple data set format levels. Since there are minimum release level requirements involved, there are migration, coexistence, and fall back considerations for your sysplex configuration.

The following table lists the LOGR couple data set format levels, the minimum release level that systems must be running in order to use the LOGR couple data set, and the options required to establish each LOGR couple data set format level.

<i>Table 26. LOGR CDS format levels</i>			
LOGR CDS format level	Minimum release level	Created when IXCL1DSU utility run on release level with option	Notes
HBB5520 (also referred to as the Base format level)	MVS/ESA SP520	any pre-OS/390 V1R3 release level (that is MVS/ESA SP520)	Refer to item names LSR,LSTRR.
HBB6603 (also referred to as the OS/390 V1R3 format level)	OS/390 V1R3	any OS/390 V1R3 or higher release level through OS/390 V2R10 and z/OS V1R1, or any z/OS V1R2 or higher release level through z/OS V2R1 without option NUMBER(1) for item name SMDUPLEX, or any z/OS V1R2 or higher release level with option NUMBER(0) specified for the item name SMDUPLEX	Refer to additional item names DSEXTENT, SMDUPLEX.
HBB7705 (also referred to as the z/OS V1R2 format level)	z/OS V1R2	any z/OS V2R2 or higher release level when the ITEM NAME(SMDUPLEX) specification is not provided, or any z/OS V1R2 or higher release level with option NUMBER(1) specified for the item name SMDUPLEX	Refer to item name SMDUPLEX If the ITEM NAME(SMDUPLEX) NUMBER(1) specification is not provided on z/OS V2R2 or a higher release level, then the default format level is HBB7705. If the ITEM NAME(SMDUPLEX) NUMBER(1) specification is not provided on z/OS V1R2 or a higher release level through z/OS V2R1, then the default format level is HBB6603.

The following table lists significant system logger functions and identifies the minimum LOGR CDS format level and release level requirements.

<i>Table 27. Considerations for supporting key system logger functions</i>			
Functions	Minimum LOGR CDS format level	Minimum release level	Notes
• Log streams supported in sysplex (coupling facility list structure type).	HBB5520	MVS/ESA SP520	Support for coupling facility list structure type log streams.

Table 27. Considerations for supporting key system logger functions (continued)

Functions	Minimum LOGR CDS format level	Minimum release level	Notes
<ul style="list-style-type: none"> Managing the entry-to-element ratio (effective average buffer size) for a structure dynamically based on actual CF structure usage. Using DSEXTENT records for expanding the amount of data set directory space available for log streams (removed the 168 data set limit per log stream). Using the RETPD and AUTODELETE log stream options to aid in archiving log data and manage the amount of data kept in a log stream. Automatically allocating temporary staging data sets to safeguard log data when there is an error that leaves the only copy of log data in a volatile configuration. Associating a log stream resource manager for monitoring log data flow for recovery purposes. 	HBB6603	OS/390 V1R3	
<ul style="list-style-type: none"> Log stream type DASD-only. 	HBB5520	OS/390 V2R4	Refer to DASDONLY type log stream designation.
<ul style="list-style-type: none"> Allow avoidance of recalling migrated log stream data sets during offload 	HBB6603	OS/390 V2R8 with PTFs	Refer to log stream attribute OFFLOADRECALL.
<ul style="list-style-type: none"> Logger support for system-managed duplexing rebuild. 	HBB7705	z/OS V1R2	Refer to log stream attribute LOGGERDUPLEX.
<ul style="list-style-type: none"> Allow pending updates for most log stream attributes. Extended high level qualifier (EHLQ) for log stream data sets. 	HBB7705	z/OS V1R3	Most log stream attributes. Refer to log stream attribute EHLQ.
<ul style="list-style-type: none"> Force disconnect or deletion of a log stream. 	HBB7705	z/OS V1R7	Refer to SETLOGR FORCE command.

Table 27. Considerations for supporting key system logger functions (continued)

Functions	Minimum LOGR CDS format level	Minimum release level	Notes
<ul style="list-style-type: none"> Log stream production and test groups. Allowing a log stream name to be renamed. 	HBB7705	z/OS V1R8	Refer to log stream GROUP attribute and update NEWSTREAMNAME.
<ul style="list-style-type: none"> Automatically apply log stream duplex pending updates following a successful coupling facility user-managed structure rebuild. 	HBB7705	z/OS V1R10	Refer to log stream updates for attributes STG_DUPLEX, DUPLEXMODE, LOGGERDUPLEX.
<ul style="list-style-type: none"> Providing z/OS log stream client support. 	HBB7705	z/OS V1R13 with PTFs	Refer to log stream attribute ZAI.
<ul style="list-style-type: none"> Providing log stream primary (interim) storage consumption alert messages. 	HBB7705	z/OS V2R1	Refer to log stream attribute WARNPRIMARY.
Managing log stream advanced-current offload data sets	HBB7705	z/OS V2R2	Refer to log stream attribute LS_ALLOCAHEAD.

- IBM recommends you use the highest format level LOGR couple data set that can be used by the lowest system release level in the sysplex. This will allow for the latest system logger features to be available given the sysplex configuration.

For example, assume the lowest system release level in your sysplex is capable of using the HBB7705 format level version LOGR couple data set, but the sysplex is actually using a HBB6603 format level. This situation can occur when the LOGR couple data set is formatted on a release earlier than z/OS V1R12 without including the ITEM NAME(SMDUPLEX) NUMBER (1) specification. However, when the couple data set is formatted with this specification included, an HBB7705 format level version is established. When this HBB7705 format level LOGR couple data set enters the sysplex as the primary couple data set, system logger can provide all the functions as described. Therefore, in this example, it is beneficial for the installation to use an HBB7705 format level LOGR couple data set even if it does not intend to use System Managed Duplexing for the Coupling Facility structures that system logger uses.

- Once a particular version of a LOGR couple data set is in use as the primary or alternate, any systems at a level below the minimum required release level for the couple data set version will be unable to access the LOGR couple data set. The inability to access the LOGR couple data set means that those systems can not exploit any LOGR function, such as updating logger inventory (policy) information or connecting to any log streams.
- Once a version of a LOGR couple data set that is formatted at a higher level is brought into the sysplex, the sysplex can not revert to a LOGR couple data set that was formatted at a lower version without a sysplex-wide IPL. The sysplex will not permit a data set formatted for less capability to become the alternate LOGR couple data set.
 - The following SETXCF command to identify an alternate couple data set, will be rejected if *alternatedsname* is not also a LOGR couple data set formatted at the same or higher functional level as the primary LOGR couple data set.

```
SETXCF COUPLE,TYPE=LOGR,ACUPLE=(alternatedsname,alternatevolume)
```

- The first system that IPLs into a sysplex can not specify in COUPLExx a version of a primary LOGR couple data set that has a higher formatted level and a version of an alternate LOGR couple data set that has a lower format level. The sysplex will reject the alternate and will operate without an alternate until a version of a LOGR couple data set that has been formatted at the same or higher format level as the primary is made available as the alternate LOGR couple data set.

See [“Migration steps to enable system-managed duplexing of logger structures”](#) on page 288 for migration steps that you must take to enable system-managed duplexing of Logger structures.

LOGR couple data set use considerations

See the topics on [“Format the sysplex scope LOGR couple data set and make it available to the sysplex”](#) on page 276 and [“When the coupling facility survives the sysplex failure”](#) on page 302 for additional considerations on LOGR CDS and CF structure use.

- If you are planning to activate a newly formatted LOGR CDS as part of the sysplex-wide IPL and this is the first IPL for the sysplex, then refer to the instructions in [“Format the sysplex scope LOGR couple data set and make it available to the sysplex”](#) on page 276.
- If this is not the first IPL for the sysplex, and on the next sysplex-wide IPL you plan to continue to use a LOGR CDS (or a mirror copy of one) that had been the last one used for this sysplex, then ensure the COUPLExx parmlib member specifies the current LOGR CDS before performing the sysplex-wide IPL or respond accordingly to message IXC289D during the initial re-IPL. See *z/OS MVS System Messages, Vol 10 (IXC-IZP)* for more details.

Note: The IXC289D message might have an automated reply, such as by being in the default AUTORxx parmlib member, which might not allow the operator enough time to input a reply. Therefore, carefully consider whether automating the reply to the IXC289D message as the default is appropriate for your sysplex.

- If this is not the first IPL for the sysplex, and on the next sysplex-wide IPL you are planning to activate a newly formatted LOGR CDS, then IBM recommends you take additional steps prior to the sysplex-wide IPL. Cause all applications using the Coupling Facility structure based log streams to be disconnected and the structures used by Logger to be unallocated before re-IPLing.

Note: Partitioning all of the systems for a sysplex-wide IPL can leave the structures used by Logger still allocated with a FAILED-PERSISTENT connection. The goal for this case is to avoid having any structures used by Logger with FAILED-PERSISTENT connections.

A DISPLAY LOGGER,L command will show which log streams have connections to systems in the sysplex as well as naming any associated CF structure. The DISPLAY XCF,STR command will show which structures are allocated within the sysplex. You can then issue a DISPLAY XCF,STR,STRNAME=structname command for any structure that is still allocated by Logger. At the bottom of the command response is the connection status. You can issue SETXCF FORCE,CON,STRNAME=structname,CONNAME=IXGLOGR_XXXX to force all of the connections and deallocate a structure.

- If this is not the first IPL for the sysplex, and on the next sysplex-wide IPL you plan to activate a LOGR CDS (or a mirror copy of one) that had been previously used but this LOGR CDS is not the last one used for the sysplex, then additional steps prior to the sysplex-wide IPL are required. Cause all applications using the Coupling Facility structure based log streams to be disconnected and the structures used by Logger to be unallocated before re-IPLing (refer to the actions in the previous bullet).

For this case, if after the sysplex-wide IPL any structures that Logger would use still have FAILED-PERSISTENT connections, then unpredictable results may occur. The problems can result in conflicts between the LOGR CDS state information for log stream resources and the state information that may also be kept in the CF structures. When the LOGR CDS log stream state information indicates there are no outstanding connections to the log stream, then the CF structure lists are cleared before Logger will allow the next log stream connection to complete. If the LOGR CDS log stream state information indicates there were outstanding (failed-persistent) connections to the log stream, then conflicts could exist between the LOGR CDS state data and any state data kept in the structure lists. The conflicting state data can lead to unpredictable results for Logger's use of the log stream or structure resources.

For example, the log stream may become unusable, loss of data conditions, or log stream offload errors can result.

LOGRY or LOGRZ single-system scope parameters for format utility

For a single-system scope policy - DATA TYPE(**LOGRY**) or DATA TYPE(**LOGRZ**), valid data names are LSR, DSEXTENT, and FMTLEVEL.

When formatting the couple data sets to support either the system logger single-system scope LOGRY or LOGRZ policy data, IBM recommends that the value specified for MAXSYSTEM should match the value specified for MAXSYSTEM when formatting the sysplex couple data sets.

ITEM NAME(LSR) NUMBER()

Specifies the maximum number of log streams that can be defined to either the LOGRY or LOGRZ single-system policy.

Each LSR record consumes one track in the LOGRY or LOGRZ couple data set.

(Default=1, Minimum=1, Maximum=32767)

ITEM NAME(DSEXTENT) NUMBER()

Specifies the number of additional log stream data set directory extents to define for log stream offload data sets. Use this parameter to increase the potential log data set directory capacity for a sysplex.

Each additional directory extent specified goes into a common pool available to any log stream in the sysplex. System logger allocates these directory extents as needed when a log stream runs out of DASD log stream data set directory space. Each directory extent allows a log stream to extend by 168 log data sets.

When a DSEXTENT record is assigned to a log stream, that particular DSEXTENT record is only used for that specific log stream until it is no longer needed. Logger then removes the DSEXTENT record from that specific log stream use and puts the DSEXTENT record back into the common pool.

Each DSEXTENT record consumes one track in the single-system (LOGRY or LOGRZ) couple data set.

(Default=0, Minimum=0, Maximum=99998)

For guidance on the DSEXTENT parameter, see [“Increasing the space available for DASD log data sets” on page 262.](#)

ITEM NAME(FMTLEVEL) NUMBER()

Specifies the format level of the single-system scope LOGRY or LOGRZ couple data set. For an explanation of the different versions or levels of the single-system LOGRY or LOGRZ couple data sets and the system logger functions that are supported, see [“LOGRY or LOGRZ couple data set versioning - new format levels” on page 327.](#)

The value of the NUMBER parameter indicates one of the following:

Value	Description
1	The resulting format level of the respective single-system scope LOGRY or LOGRZ couple data set is HBB77C0 and is supported on z/OS V2R4 and later releases.

When this item is omitted from the IXCL1DSU format utility, logger treats it as if a value of one (1) was specified.

This specification does not consume any space in the respective LOGRY or LOGRZ couple data set.

(Default=1, Minimum=1, Maximum=1)

LOGRY or LOGRZ couple data set versioning - new format levels

For general couple data set coexistence considerations, refer to [“Considerations for all couple data sets”](#) on page 27 and [“Understanding couple data set coexistence”](#) on page 313.

Logger allows format levels or functional versions of single-system scope LOGRY or LOGRZ couple data sets to be defined. Certain logger functions require a specific format level of the LOGRY or LOGRZ couple data set as well as a particular release level of z/OS.

Table 28 on page 327 lists the single-system scope LOGRY and LOGRZ couple data set format levels, the minimum release level that systems must be running in order to use the single-system scope LOGRY or LOGRZ couple data set, and the options required to establish each LOGRY or LOGRZ couple data set format level.

<i>Table 28. LOGRY and LOGRZ CDS format levels</i>			
LOGRY or LOGRZ CDS format level	Minimum release level	Created when IXCL1DSU utility run on release level with option	Notes
HBB77C0 Also referred to as the Base format level for the single-system scope CDS types.	z/OS V2R4	Any z/OS V2R4 or higher release level (or any z/OS V2R2 or z/OS V2R3 release level with PTFs for APAR OA54815 applied using the STEPLIB parameter to reference an up-level library) and with NUMBER(1) specified for the item name FMTLEVEL or when the FMTLEVEL item name is omitted.	Refer to item names LSR, DSEXTENT and FMTLEVEL.

Table 29 on page 327 lists significant system logger functions and identifies the minimum single-system scope LOGRY or LOGRZ CDS format level and release level requirements.

<i>Table 29. Considerations for supporting key system logger functions</i>			
Functions	Minimum LOGRY or LOGRZ format level	Minimum release level	Notes
All the system logger functions supported for a sysplex scope LOGR CDS data type at the z/OS V2R4 release level, except for functions related to using coupling facility list structure type log streams.	HBB77C0	z/OS V2R4	Only log stream type DASD-only are supported with single-system scope LOGRY or LOGRZ couple data sets.

IBM recommends that you use the highest format level LOGRY and LOGRZ couple data set that can be used by the lowest system release level in the sysplex. This will allow for the latest system logger features to be available given the system release level and sysplex configuration.

Formatting a sysplex scope LOGR couple data set

Sample JCL to run the format utility for formatting sysplex scope couple data sets for the system logger service is shipped in SYS1.SAMPLIB member IXGLOGRF.

The sample shows formatting for both a primary and alternate couple data set for the LOGR policy.

In the sample JCL:

```

STEP 1 defines two LOGR couple data sets at the (recommended)
HBB7705 format level with enough room to allow:
  10 logstreams to be defined
  10 logger structures to be defined
  20 DSEXTENT records defined for extending log stream
  directory space for offload data sets
STEP 2 defines two LOGR couple data sets at the
HBB6603 format level with enough room to allow:
  10 logstreams to be defined
  10 logger structures to be defined
  20 DSEXTENT records defined for extending log stream
  directory space for offload data sets

```

Figure 69. Example of formatting a primary and alternate couple data set for the LOGR policy

See [“LOGR couple data set versioning — new format levels”](#) on page 321 for more information about the different LOGR CDS format levels.

Note: If you want the LOGR policy to be in an SMS-managed volume, specify the storage and management classes on the DEFINEDS statement. The parameter for storage class is STORCLAS. The parameter for management class is MGMTCLAS. For detailed information, see the description of the DEFINEDS parameter and also the topic of a couple data set can be defined on an sms-managed volume.

Formatting a single-system scope LOGRY or LOGRZ couple data set

Sample JCL to run the format utility for formatting single-system scope couple data sets for the system logger service is shipped in SYS1.SAMPLIB member IXGLOGZF.

The sample shows formatting for both a primary and alternate couple data set for both the LOGRY policy and LOGRZ policy.

Note: If you want either the LOGRY or LOGRZ policy to be in an SMS-managed volume, specify the storage and management classes on the DEFINEDS statement. The parameter for storage class is STORCLAS. The parameter for management class is MGMTCLAS.

For detailed information, see 'A Couple Data Set Can Be Defined on an SMS-Managed Volume' in [“Considerations for all couple data sets”](#) on page 27.

SFM parameters for format utility

For an SFM couple data set — DATA TYPE(SFM), valid data names are POLICY, SYSTEM, and RECONFIG.

ITEM NAME(POLICY) NUMBER()

Specifies the number of administrative policies that can be defined. (Default=9, Minimum=1, Maximum=50)

ITEM NAME(SYSTEM) NUMBER()

Specifies the number of systems for which actions and weights can be defined. This should be the maximum number of systems that will be in the sysplex that the policy will govern. Note that the number specified does not need to include those systems identified by NAME(*), for which policy default values are applied. (Default=8, Minimum=0, Maximum=32)

ITEM NAME(RECONFIG) NUMBER()

Specifies the number of reconfigurations involving PR/SM partitions that can be specified. (Default=0, Minimum=0, Maximum=50)

z/OS UNIX parameters for format utility

For an OMVS couple data set — DATA TYPE(BPXMCDs), valid data names are MOUNTS and AMTRULES.

ITEM NAME(MOUNTS) NUMBER()

Specifies the number of MOUNTS that can be supported by z/OS UNIX. (Default=100, Minimum=1, Maximum=50000)

ITEM NAME(AMTRULES) NUMBER()

Specifies the number of automount rules that can be supported by z/OS UNIX. (Default=50, Minimum=50, Maximum=5000)

See [“Tuning UNIX System Services performance in a sysplex” on page 166](#) for information about using a shared file system in a sysplex.

The NUMBER (nnnn) specified for mounts and automount rules (a generic or specific entry in an automount map file) is directly linked to function performance and the size of the OMVS couple data set. If maximum values are specified, the size of the couple data set will increase accordingly and the performance level for reading and updating it will decline. Conversely, if the NUMBER values are too small, the function (for example, the number of mounts supported) would fail once the limit is reached. However, if such a failure occurs, a new couple data set can be formatted and switched in with larger values specified for NUMBER. The number of file systems required (factoring in an additional number to account for extra mounts), will determine your minimum and maximum NUMBER value.

BPXMCDS couple data set versioning – new format level

z/OS UNIX allows different levels or functional versions of OMVS couple data sets to be defined. The following couple data set format levels or functional versions can be created using the IXCL1DSU utility.

- Base format level (Version 1). This is the initial or base z/OS UNIX couple data set format level and is created when the OMVS couple data set is formatted using a version of IXCL1DSU prior to z/OS V1R4.
- z/OS V1R4 format level (Version 2). This format level is created when the OMVS couple data set is formatted using a version of IXCL1DSU at z/OS V1R4 or higher.

OMVS couple data sets formatted with the z/OS V1R4 version of IXCL1DSU are required in order to use the enhanced shared file system algorithms that are available in V1R4. Systems at different system levels can access the V1R4-formatted z/OS UNIX couple data set as described in the following scenario.

After IPL-ing the z/OS V1R4 system, initialization processing for the shared file system function in the sysplex will be delayed until an OMVS couple data set formatted at the z/OS V1R4 level is made available as the primary couple data set.

1. Using the z/OS V1R4 version of IXCL1DSU, define new primary and alternate OMVS couple data sets.
2. Enable the new Version 2 couple data set as the primary couple data set as follows:

- If there is currently no primary OMVS couple data set enabled in the sysplex, use the SETXCF COUPLE system command to identify one. For example,

```
SETXCF COUPLE,TYPE=BPXMCDS,PCOUPLE=(POSIX.OMVS.CDS01,OMVSDS)
```

- Alternately, if there is already a primary OMVS couple data set enabled in the sysplex (by another active system in the sysplex), use the SETXCF COUPLE system command to enable the version 2 couple data set as the alternate couple data set and then use the SETXCF PSWITCH system command to switch the alternate couple data set to the primary couple data set.

```
SETXCF COUPLE,TYPE=BPXMCDS,PSWITCH
```

These commands should be issued from a system where the OMVS couple data set is enabled as the primary. Enabling a version 2 OMVS couple data set as the primary couple data set is transparent to other active systems in the sysplex that are configured for shared file systems in a sysplex.

Once the version 2 OMVS couple data set is enabled as the primary couple data set in the sysplex, shared file system initialization processing on the z/OS V1R4 system should resume.

WLM parameters for format utility

For a WLM couple data set — DATA TYPE(WLM), valid data names are POLICY, WORKLOAD, SRVCLASS, SVDEFEXT, SVDCREXT, APPLENV, SVAEAEXT, SCHENV, and SVSEEXT.

ITEM NAME(POLICY) NUMBER()

Specifies that an increment of space large enough to accommodate the specified number of policies be allocated in the WLM couple data set. (Default=5, Minimum=1, Maximum=99)

ITEM NAME(WORKLOAD) NUMBER()

Specifies that an increment of space large enough to accommodate the specified number of workloads be allocated in the WLM couple data set. (Default=32, Minimum=1, Maximum=999)

ITEM NAME(SRVCLASS) NUMBER()

Specifies that an increment of space large enough to accommodate the specified number of service classes be allocated in the WLM couple data set. (Default=128, Minimum=1, Maximum=999)

Note: WLM allows no more than 100 service classes to be defined in a service definition. The default, however, remains at the value of 128. This will set aside as much space as you will ever need for service classes, as well as a little extra for other WLM objects.

ITEM NAME(SVDEFEXT) NUMBER()

Specifies that an exact amount of space (in K bytes) for extension areas to the WLM Service Definition (IWMSVDEF) be allocated in the WLM couple data set. (Default=0, Minimum=0, Maximum=8092)

ITEM NAME(SVDCREXT) NUMBER()

Specifies that an exact amount of space (in K bytes) for extension areas to the WLM Service Definition Classification Rules (IWMSVDCR) be allocated in the WLM couple data set. (Default=0, Minimum=0, Maximum=8092)

ITEM NAME(APPLENV) NUMBER()

Specifies that an increment of space large enough to accommodate the specified number of application environments be allocated in the WLM couple data set. (Default=100, Minimum=1, Maximum=3000)

ITEM NAME(SVAEEXT) NUMBER()

Specifies that an exact amount of space (in K bytes) for extension areas to the WLM Service Definition Application Environment Area (IWMSVAEA) be allocated in the WLM couple data set. (Default=0, Minimum=0, Maximum=8092)

ITEM NAME(SCHENV) NUMBER()

Specifies that an increment of space large enough to accommodate the specified number of scheduling environments be allocated in the WLM couple data set. (Default=100, Minimum=1, Maximum=999)

ITEM NAME(SVSEAEXT) NUMBER()

Specifies that an exact amount of space (in K bytes) for extension areas to the WLM Service Definition Scheduling Environment Area (IWMSVSEA) be allocated in the WLM couple data set. (Default=0, Minimum=0, Maximum=8092)

Return codes

The return codes in decimal format from the format utility are:

Return Codes**Description****0**

Successful completion. All requested data sets are formatted.

4

One or more couple data sets could not be formatted.

8

An I/O error occurred on the SYSPRINT or SYSIN file. Some couple data sets might have been formatted.

12

The SYSPRINT DD or SYSIN DD was not specified or could not be used. No data sets are formatted.

16

An abend occurred during format utility processing. Some couple data sets might have been formatted.

20

Unknown error while parsing command input. Some couple data sets might have been formatted.

Examples of using the IXCL1DSU format utility

The examples shown here for z/OS are available in SYS1.SAMPLIB.

Formatting a sysplex couple data set

Sample JCL to run the format utility to format a sysplex couple data set is shipped with z/OS in SYS1.SAMPLIB member IXCSYSPF.

Formatting an automatic restart management couple data set

Sample JCL to run the format utility for format couple data sets for automatic restart management is shipped in SYS1.SAMPLIB member IXCARMF.

Formatting a CFRM couple data set

Sample JCL to run the format utility for formatting couple data sets for the coupling facility resource management service is shipped in SYS1.SAMPLIB member IXCCFRMF.

Formatting a sysplex scope LOGR couple data set

Sample JCL to run the format utility for formatting sysplex scope couple data sets for the system logger service is shipped in SYS1.SAMPLIB member IXGLOGRF.

The sample shows formatting for both a primary and alternate couple data set for the LOGR policy.

In the sample JCL:

```
STEP 1 defines two LOGR couple data sets at the (recommended)
HBB7705 format level with enough room to allow:
  10 logstreams to be defined
  10 logger structures to be defined
  20 DSEXTENT records defined for extending log stream
  directory space for offload data sets
STEP 2 defines two LOGR couple data sets at the
HBB6603 format level with enough room to allow:
  10 logstreams to be defined
  10 logger structures to be defined
  20 DSEXTENT records defined for extending log stream
  directory space for offload data sets
```

Figure 70. Example of formatting a primary and alternate couple data set for the LOGR policy

See [“LOGR couple data set versioning — new format levels”](#) on page 321 for more information about the different LOGR CDS format levels.

Note: If you want the LOGR policy to be in an SMS-managed volume, specify the storage and management classes on the DEFINEDS statement. The parameter for storage class is STORCLAS. The parameter for management class is MGMTCLAS. For detailed information, see the description of the DEFINEDS parameter and also the topic of a couple data set can be defined on an sms-managed volume.

Formatting an SFM couple data set

Sample JCL to run the format utility for formatting couple data sets for the sysplex failure management service is shipped in SYS1.SAMPLIB member IXCSFMF and is shipped in SYS1.SAMPLIB member IXCSFMF.

Formatting a UNIX System Services couple data set

Sample JCL to run the format utility to format a UNIX System Services couple data set is shipped with z/OS in SYS1.SAMPLIB member BPXISCDS. It allows for specification of the number of mount records that will be supported by the couple data set.

Formatting a WLM couple data set

Sample JCL to run the format utility for formatting a primary couple data set for workload management is shipped in SYS1.SAMPLIB member IWMFTCDS.

Sample output

[Figure 71 on page 333](#) input statements and output messages from IXCL1DSU formatting a single couple data set for CFRM, SFM, WLM, ARM and LOGR.

```

DEFINEDS SYSPLEX(XLSDEV)
  DSN(SYS1.XCF.FDS01) VOLSER(3380X1)
  MAXSYSTEM(32)
  CATALOG
  DATA TYPE(CFRM)
    ITEM NAME(POLICY)    NUMBER(6)
    ITEM NAME(CF)        NUMBER(5)
    ITEM NAME(STR)        NUMBER(20)
    ITEM NAME(CONNECT)    NUMBER(32)
  DATA TYPE(SFM)
    ITEM NAME(POLICY)    NUMBER(9)
    ITEM NAME(SYSTEM)    NUMBER(8)
    ITEM NAME(RECONFIG)   NUMBER(4)
  DATA TYPE(WLM)
    ITEM NAME(POLICY)    NUMBER(10)
    ITEM NAME(WORKLOAD)   NUMBER(35)
    ITEM NAME(SRVCLASS)   NUMBER(30)
    ITEM NAME(SVDEFEXT)   NUMBER(5)
    ITEM NAME(SVDCREXT)   NUMBER(5)
    ITEM NAME(APPLENV)    NUMBER(100)
    ITEM NAME(SVAEAEXT)   NUMBER(5)
    ITEM NAME(SCHENV)     NUMBER(100)
    ITEM NAME(SVSEAEXT)   NUMBER(5)
  DATA TYPE(ARM)
    ITEM NAME(POLICY)    NUMBER(5)
    ITEM NAME(MAXELEM)    NUMBER(25)
    ITEM NAME(TOTELEM)    NUMBER(20)
  DATA TYPE(LOGR)
    ITEM NAME(LSR)        NUMBER(50)
    ITEM NAME(LSTRR)      NUMBER(20)

IGD100I 0182 ALLOCATED TO DDNAME COUPLE  DATACLAS (      )
IXC292I DATA SET FORMATTING COMPLETE: DATA SET REQUIRES 208 TRACKS ON VOLSER
3380X1
IXC292I 1 RECORDS FORMATTED WITH 6 POLICY ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 5 CF ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 20 STR ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 32 CONNECT ITEMS EACH

IXC292I 1 RECORDS FORMATTED WITH 9 POLICY ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 8 SYSTEM ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 4 RECONFIG ITEMS EACH

IXC292I 10 POLICY RECORDS FORMATTED
IXC292I 35 WORKLOAD RECORDS FORMATTED
IXC292I 30 SRVCLASS RECORDS FORMATTED
IXC292I 5 SVDEFEXT RECORDS FORMATTED
IXC292I 5 SVDCREXT RECORDS FORMATTED
IXC292I 100 APPLENV RECORDS FORMATTED
IXC292I 5 SVAEAEXT RECORDS FORMATTED
IXC292I 100 SCHENV RECORDS FORMATTED
IXC292I 5 SVSEAEXT RECORDS FORMATTED

IXC292I 1 RECORDS FORMATTED WITH 20 TOTELEM ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 5 POLICY ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 25 MAXELEM ITEMS EACH

IXC292I 1 RECORDS FORMATTED WITH 20 LSR ITEMS EACH
IXC292I 1 RECORDS FORMATTED WITH 50 LSTRR ITEMS EACH

```

Figure 71. Sample Output of Couple Data Set Format Utility

Chapter 12. Administrative data utility

Use the administrative data utility to add, update, or delete policy data on the formatted ARM, CFRM, LOGR, LOGRY, LOGRZ and SFM couple data sets. The number of administrative policies you can add is determined by the number specified on the POLICY control statement in the couple data set format utility.

Note to WLM Users: The administrative data utility does not support WLM policy information. Those policies are stored in a formatted WLM couple data set with an ISPF application. See [z/OS MVS Planning: Workload Management](#) for a description of the ISPF application and how to use it.

The control statements for this utility program follow standard conventions for JCL statements. The utility accepts 80-byte records with one or more parameters per record. See [z/OS MVS JCL Reference](#) for additional information about JCL statements.

The administrative data utility is portable; that is, you can run the utility on any system running any version of z/OS.

Using the administrative data utility

The name of the administrative data utility is IXCMIAPU. The program resides in SYS1.MIGLIB (which is logically appended to the LINKLIST), which makes it available through STEPLIB on a non-OS/390 MVS system.

Programs that use the IXCMIAPU utility must reside in an APF-authorized library.

Note: By default SYS1.MIGLIB is APF-authorized.

See [z/OS MVS Initialization and Tuning Reference](#) for information about the LNKAUTH parameter in IEASYSxx and about defining the APF list. Note that if you want to run a different system's level of IXCMIAPU, that system's MIGLIB must be APF-authorized on the system where the utility is to be run.

Authorizing use of the utility

You must control the use of the IXCMIAPU utility through the z/OS Security Server, which includes RACF, or your installation's security package. The security administrator must define a resource profile for the resource name associated with the following policies if the installation plans to use them:

- For automatic restart management, the resource name is 'MVSADMIN.XCF.ARM'.
- For CFRM, the resource name is 'MVSADMIN.XCF.CFRM'.
- For LOGR, the resource name is 'MVSADMIN.LOGR'.
- For LOGRY, the resource name is 'MVSADMIN.LOGRY'.
- For LOGRZ, the resource name is 'MVSADMIN.LOGRZ'.
- For SFM, the resource name is 'MVSADMIN.XCF.SFM'.

If you are using the z/OS Security Server (RACF), define the FACILITY class profiles shown in [Figure 72 on page 335](#).

```
RDEFINE FACILITY MVSADMIN.XCF.ARM UACC(NONE)
RDEFINE FACILITY MVSADMIN.XCF.CFRM UACC(NONE)
RDEFINE FACILITY MVSADMIN.LOGR UACC(NONE)
RDEFINE FACILITY MVSADMIN.LOGRY UACC(NONE)
RDEFINE FACILITY MVSADMIN.LOGRZ UACC(NONE)
RDEFINE FACILITY MVSADMIN.XCF.SFM UACC(NONE)
```

Figure 72. FACILITY Class Profiles

Assign UPDATE access authority to users who must alter or maintain the policy; assign READ access authority to users who require reports on the policy, but who will not change the policy. (These FACILITY

class profiles will be used for authority checking on an ACTIVE couple data set. When a couple data set is INACTIVE, the rules for normal data set protection apply.)

After giving access authority to the appropriate users, activate the FACILITY class.

```
SETROPTS CLASSACT(FACILITY)
```

Coding the administrative data utility

Use the administrative data utility to add, update, or delete policy data on the formatted ARM, CFRM, LOGR, LOGRY, LOGRZ and SFM couple data sets. The number of administrative policies you can add is determined by the number specified on the POLICY control statement in the couple data set format utility.

For examples of how to set up your JCL and code the control statements for administrative policy data, see the following figures:

- For automatic restart management, see [“Administrative data utility for automatic restart manager Policy Data”](#) on page 397.
- For CFRM, see [“Administrative data utility for CFRM policy data”](#) on page 398.
- For LOGR, see [“Administrative data utility for the LOGR policy”](#) on page 398.
- For LOGRY and LOGRZ, see [“Administrative data utility for the LOGRY and LOGRZ policy data”](#) on page 399.
- For SFM, see [“Administrative data utility for SFM policy data”](#) on page 399.

The utility control statements are described below:

PGM=IXCMIAPU

The administrative data utility program that is shipped with SP 5.1. (Program IXCM2APU also exists as an alias for IXCMIAPU.)

SYSPRINT

Describes where the output messages from the administrative data utility are to be printed. The SYSPRINT DD statement is required.

SYSABEND

Describes where SYSABEND data from the administrative data utility is to be printed. The SYSABEND DD statement is required.

SYSIN

Describes the utility control statements that are input to the utility.

DATA

Indicates that a couple data set is to be updated. The couple data set is identified with the following parameters of DATA.

TYPE(ARM)

TYPE(CFRM)

TYPE(LOGR)

TYPE(LOGRY)

TYPE(LOGRZ)

TYPE(SFM)

Specifies the type of administrative data that the data set is to contain. TYPE is a required parameter.

LOGR , LOGRY and LOGRZ Policy-Specific Information:The information for the system logger type of policy being updated is identified with specific parameters of the DATA statement.

See [“LOGR keywords and parameters for the administrative data utility”](#) on page 358 for a description of the keywords to identify log streams and structures in a sysplex scope LOGR policy.

See [“LOGRY and LOGRZ keywords and parameters for the administrative data utility”](#) on page 386 for a description of the keywords to identify DASD-only log streams in a single-system scope LOGRY or LOGRZ policy.

DSN(data-set-name)

Specifies the 1 - 44 character length name of the couple data set that was formatted with the IXCL1DSU format utility to contain the specified type of administrative data. The valid characters are alphanumeric characters (A-Z and 0-9), national characters (\$,@,#), a period (.), and a hyphen (-). The first character must be either an alphabetic or a national character.

DSN is an optional parameter when you are updating administrative data. When you do not specify DSN when updating administrative data, the utility applies the changes to the data set (or data sets, if both primary and alternate exist) that contains the active policy — CFRM, SFM, or ARM, and any administrative policies.

The DSN parameter is not supported for TYPE(LOGR), TYPE(LOGRY) or TYPE(LOGRZ), since you can only update the active system logger administrative policy. If you specify DSN for one of the system logger couple data set type policies, the IXCMIAPU utility ends with a non-zero return code and error message IXG432E.

Note that the changes are applied to the administrative policy. If the changes are to an administrative policy that is also the current active policy, the changes are applied to the administrative copy of the active policy and not directly to the active policy itself. To activate the changes in the active policy in the sysplex, issue a SETXCF command to make the updated administrative policy the active policy.

If you want to obtain a report of the policy information contained in the active couple data set, ensure that you omit the DSN keyword. Otherwise, the system issues allocation error messages.

When to Specify DSN: (An active couple data contains the current active policy.)

- DO specify DSN if you are creating your administrative policies initially or want to have a secondary set of policies in a couple data set that is not active in your sysplex.
- DO NOT specify DSN if you want to add a new policy or update an existing policy in the couple data set that is active in your sysplex.

See [“Updating a CFRM policy” on page 66](#) for an example of how to update a policy.

VOLSER(serial number)

Specifies the 1 - 6 character length name of the volume on which the couple data set resides if the data set is not cataloged. The valid characters are alphanumeric characters (A-Z and 0-9) and national characters (\$,@,#). VOLSER is an optional parameter and can be specified only if DSN parameter also is specified.

The VOLSER parameter is not supported for TYPE(LOGR), TYPE(LOGRY) or TYPE(LOGRZ), since you can only update the active system logger administrative policy. If you specify VOLSER for one of the system logger couple data set type policies, the IXCMIAPU utility ends with a non-zero return code and error message IXG432E.

REPORT(YES)**REPORT(NO)**

Specifies whether a report of the contents of the policy data for all administrative policies defined in the data set is to be output to the SYSPRINT file. The default is YES, to output a report.

Note: If you specify REPORT(YES) (or accept the default for the report keyword) with a DATA TYPE of LOGR, the requestor must have SAF read access to the MVSADMIN.LOGR resource to successfully obtain a report. For a DATA TYPE of LOGRY or LOGRZ, the requestor must have respective SAF read access to the MVSADMIN.LOGRY or MVSADMIN.LOGRZ resource to successfully obtain a report.

DEFINE POLICY NAME(polname)

Specifies the beginning of the definition of a new administrative policy. The limit for the number of policies that can be defined in a couple data set is established when the couple data set is formatted.

polname specifies the name of the policy. The name can be from 1 - 8 characters in length. Valid characters are A-Z and 0-9. The policy name must start with an alphabetic character.

A policy name can be defined only once in a single run of the administrative data utility. **ARM, CFRM, and SFM Policy-Specific Information:** The information for the ARM, CFRM, and SFM **type** of policy being updated is identified with specific parameters of the DEFINE POLICY statement.

- See “Automatic restart management parameters for administrative data utility” on page 338 for a description of the keywords to define elements and restart groups in an automatic restart management policy.
- See “CFRM parameters for administrative data utility” on page 346 for a description of the keywords to identify coupling facilities and structures in a CFRM policy.
- See “SFM parameters for the administrative data utility” on page 392 for a description of the keywords to identify systems in the sysplex and PR/SM reconfiguration scenarios in an SFM policy.

The LOGR, LOGRY and LOGRZ policies do not use the DEFINE POLICY statement. See “LOGR keywords and parameters for the administrative data utility” on page 358 for the sysplex scope LOGR policy-specific keywords on the DATA statement. See “LOGRY and LOGRZ keywords and parameters for the administrative data utility” on page 386 for the single-system scope LOGRY and LOGRZ policy-specific on the DATA statement.

REPLACE(NO)

REPLACE(YES)

REPLACE(timestring)

Specifies whether or not the existing policy identified by *polname* is to be replaced.

- If the named administrative policy currently exists in the data set, REPLACE(YES) causes the policy definition to be replaced with the policy being defined. REPLACE(NO) will not allow the policy definition in the data set to be replaced.
- If the named administrative policy does not currently exist in the data set, either REPLACE(YES) or REPLACE(NO) allow the policy definition to be added to the data set.
- If you specify REPLACE(*timestring*), the system will only replace policy *polname* if the policy is already defined in the CFRM couple data set with a matching time string, which is the local time of the policy definition. *timestring* can be up to 26 characters long (not including the quotation marks) in the format '*mm/dd/yyyy hh:mm:ss.ssssss*'.

You can obtain the the policy definition time by running the administrative policy utility with REPORT(YES) specified.

You can only specify REPLACE(*timestring*) for the CFRM policy.

DELETE POLICY NAME(polname)

Specifies that the named administrative policy is to be deleted from the set of policies previously defined by the administrative data utility on the couple data set.

Automatic restart management parameters for administrative data utility

The automatic restart management policy identifies elements and restart groups with restart parameter values that differ from the policy default values.

Your installation does not need to specify a policy if the default values are acceptable, or if the application overrides the defaults when coding the IXCARM macro. The default values (if any) are described under each parameter, along with the IXCARM macro specifications that can be used to override these defaults.

Note: The automatic restart management policy values are determined when an element is restarted, not when the batch job or started task registers with automatic restart management.

RESTART_ORDER

Specifies the order in which elements in the same restart group are to become ready after they are restarted. The RESTART_ORDER parameter applies to all restart groups. RESTART_ORDER is optional. The default values for RESTART_ORDER are:

- LEVEL(0) - ELEMENT_TYPES: SYSIRLM, SYSLVLO
- LEVEL(1) - ELEMENT_TYPES: SYSDDB2, SYSIMS, SYSVTAM, SYSTCPIP, SYSLVL1

- LEVEL(2) - ELEMENT_TYPEs: SYSCICS, SYSMQMGR, SYSMQCH, SYSKERB, SYSLVL2
- LEVEL(3) - ELEMENT_TYPEs: SYSCB, SYSLVL3

Any elements that do not match a default or policy-specified ELEMENT_TYPE or a policy-specified ELEMENT_NAME.

LEVEL(level)

Specifies the level associated with elements that must be restarted in a particular order. The elements are restarted from the lowest level to the highest level. The LEVEL keyword can be specified multiple times.

The set of elements associated with a particular level is identified by the ELEMENT_NAME or ELEMENT_TYPE parameters. ELEMENT_NAME or ELEMENT_TYPE must be specified with each LEVEL specification. (You can specify both ELEMENT_NAME and ELEMENT_TYPE.)

The *level* must be a decimal number from 0 through 65535. The default is LEVEL(2). The IXCARM macro parameter that overrides the default specification is ELEMENTYPE (which allows an element to give a name or type that is associated with another LEVEL).

ELEMENT_NAME(name-list)

Specifies the name of each element to be restarted at the *level* specified by LEVEL. The element name can contain the wildcard characters '?' or '*'. See [“Using wildcard characters” on page 345](#) for more information about wildcard characters.

The *name-list* can be one or more element names separated by commas.

IBM reserves the use of element names starting with the characters A through I and "SYS" for IBM use.

ELEMENT_TYPE(type-list)

Specifies the element types that are to be restarted at the *level* specified by LEVEL. The element type is a name given to elements with similar characteristics. For example, PAYROLL might be the element type of multiple payroll-related programs (elements).

Type-list can be one or more element types separated by commas. You can specify the element type on the IXCARM macro by coding the parameter ELEMENTYPE.

IBM reserves the use of element types starting with the characters A through I and "SYS" for IBM use.

RESTART_GROUP(*|name)

Identifies related elements that are to be restarted as a group if the system on which they are running fails. Not all elements in a given restart group need to be running on the same system, or running at all.

The restart group name can:

- Be up to 16 characters long with no imbedded blanks
- Be a combination of alphabetic characters, numbers, and \$, #, and @
- Not begin with a number
- Be named DEFAULT to change the characteristics (target system, CSA requirements, and so on) of the default restart group
- Not be repeated
- Not contain wildcard characters (except for RESTART_GROUP(*))

The RESTART_GROUP parameter can be specified more than once in a policy.

However, RESTART_GROUP(*) is a special category and can be specified only once. A global substitution of the values specified here are applied to all restart groups that are not specified in the policy. For example, when the following is specified, only systems SYS1 and SYS3 can be the targets of a cross-system restart.

```
RESTART_GROUP(*)
  TARGET_SYSTEM(SYS1,SYS3)
```

As another example, the following restart groups are specified in a policy:

```
RESTART_GROUP(*)
  TARGET_SYSTEM(SYS1,SYS3)
RESTART_GROUP(JIM)
  TARGET_SYSTEM(SYS2)
  ELEMENT(CICS1)
  ELEMENT(DB28)
```

If a system fails, the elements listed in the restart group named JIM would be restarted only on SYS2. All other elements would be restarted on SYS1 or SYS3.

For each restart group, you can associate the following optional parameters:

```
RESTART_GROUP
  TARGET_SYSTEM
  FREE_CSA
  RESTART_PACING
```

Within each restart group, you specify the element or elements that are members of the restart group. Each element specifies its own set of restart requirements.

RESTART_GROUP is a required parameter. All elements that are not put into a specific restart group by a policy are in the restart group named DEFAULT.

TARGET_SYSTEM(*|system-list)

Specifies the systems on which elements can be restarted in a cross-system restart. Specify this parameter when you do not want all systems to be the targets of a cross-system restart.

The *system-list* must be one or more system names separated by commas. The order of the systems in *system-list* is irrelevant.

TARGET_SYSTEM(*) indicates that any system that supports automatic restart management might be the target of a cross-system restart. When the sysplex is running with WLM in goal mode, automatic restart management will choose the system that has the greatest available CPU capacity and that has the required available CSA (if the FREE_CSA parameter was specified for this restart group).

TARGET_SYSTEM is optional. If specified, the name of the target system must be 1-8 characters. Valid characters are alphanumeric (A-Z) and (0-9) and national characters (@,#,\$). The first character of the target system name can be an alphanumeric or national character. TARGET_SYSTEM(*) is the default.

FREE_CSA(below,above)

Specifies the minimum amount of CSA and ECSA that must be available on a system for it to be eligible for a cross-system restart of the elements in the associated restart group.

FREE_CSA is optional. The default is FREE_CSA(0,0).

Note: This parameter value will not disqualify a system as eligible for a cross-system restart if WLM is running in compatibility mode on that system.

below

Indicates the number of kilobytes (KB) of CSA that must be available on a target system for this restart group to be restarted. A value of 0 indicates that the amount of free CSA is not a requirement for eligible systems for this restart group.

The maximum decimal value that can be specified for this keyword is 16384. The default is 0.

above

Indicates the number of kilobytes (KB) of ECSA that must be available on a target system for this restart group to be restarted. A value of 0 indicates that the amount of free ECSA is not a requirement for eligible systems for this restart group.

The maximum decimal value that can be specified for this keyword is 2080767. The default is 0.

RESTART_PACING(delay)

Specifies the amount of time in seconds that automatic restart management should delay between the restart of each element in the restart group. This value provides the ability to stagger the restarts of the elements to minimize contention on a system.

The value for *delay* must be a decimal number from 0 to 21600 seconds (6 hours). This value applies only for cross-system restarts.

RESTART_PACING is optional. The default is RESTART_PACING(0).

ELEMENT(*|name)

Specifies a batch job or started task that can register as an element of automatic restart management. The wildcard characters '?' or '*' can be used. See [“Using wildcard characters”](#) on page 345 for more information about wildcard characters.

In addition, automatic restart management provides two system symbols that relate to the policy syntax for automatic restarts.

IBM reserves the use of element names starting with the characters A through I and "SYS" for IBM use.

&SYSELEM.

Specifies that the element name from the previous ELEMENT statement should be substituted for this system symbol. Element names used for &SYSELEM. cannot exceed 8 characters in length. For example:

```
ELEMENT (KAREN??)
  RESTART_METHOD (SYSTEM,STC, 'S &SYSELEM.')
```

If an element named KAREN1A needs to be restarted, MVS substitutes 'S KAREN1A' for the command text.

&SYSSUF.

Substitutes the suffix of an element name. The characters following the first wildcard character are considered the suffix.

For example, if element ELWOOD was being restarted and the best match from an installation policy was ELEMENT(EL*), then the suffix for this element would be WOOD. Similarly, when element PKASKOVICH matches the policy entry ELEMENT(PKAS??VI*), then the suffix for this element would be KOVICH.

The substitution text for &SYSSUF. cannot exceed 7 characters.

See [“Using system symbol substitution”](#) on page 345 for general information about using system symbols.

You can specify ELEMENT(*) only when RESTART_GROUP(*) or RESTART_GROUP(DEFAULT) is specified. When ELEMENT(*) is specified with RESTART_GROUP(*), the element-related parameters apply to all elements in all restart groups (if these parameters do not have explicitly specified values for those restart groups). When ELEMENT(*) is specified with RESTART_GROUP(DEFAULT), the associated parameters apply to all elements in this restart group.

ELEMENT is a required parameter. For each element specified, you can associate the following optional parameters:

```
ELEMENT
  RESTART_ATTEMPTS
  CLEANUP_TIMEOUT
  RESTART_TIMEOUT
  READY_TIMEOUT
  TERMTYPE
  RESTART_METHOD
```

RESTART_ATTEMPTS(max-number,time-interval)

Specifies the maximum number of times that automatic restart management should attempt to restart the specified element within a given interval. This limit prevents automatic restart management from continually restarting an element that is recursively terminating.

When this threshold has been reached, no further restarts will be attempted and the system will deregister the element.

max-number

Must be a decimal value from 0 to 3. When RESTART_ATTEMPTS(0) is specified, no restarts will be attempted for this element. The default is 3.

time-interval

Must be a decimal number from 1 to 86400 seconds (24 hours). The default is 300.

RESTART_ATTEMPTS is optional, but if specified, must refer to a specific ELEMENT.

The default is RESTART_ATTEMPTS(3,300).

CLEANUP_TIMEOUT(time-interval)

Specifies the maximum amount of time, in seconds, that cross-system restart of an element is to be delayed waiting for member cleanup of a terminated system.

time-interval

Must be a decimal number from 120 (2 minutes) to 86400 seconds (24 hours).

When a system terminates, automatic restart management updates the state of elements owned by that terminated system to FAILED. When member cleanup for the terminated system completes, systems targeted to perform cross-system restart update the state of eligible elements to RESTARTING and perform cross-system restart processing.

When member cleanup for the terminated system does not complete within two (120 seconds) minutes, systems that are targeted to perform cross-system restart update the state of eligible elements to RESTARTING and proceed to perform cross-system restart processing by using the CLEANUP_TIMEOUT parameter to introduce additional delay time as necessary.

The potential amount of additional delay time is any amount of time specified by the CLEANUP_TIMEOUT parameter that is more than 2 minutes. When a time-interval greater than 120 is specified (or defaulted to), and additional delay is introduced to wait for member cleanup processing to complete, message IXC815I is issued to the system log to record the delay in performing restart processing.

If the member cleanup does not occur within this interval, automatic restart management will proceed with processing the element for cross-system restart. And, if CLEANUP_TIMEOUT(120) is not being used, the system issues message IXC815I to the system log to record the time out.

Note: Cross-system restart processing performs element restart processing in order of ascending numerical level within a particular restart group. Restart processing of elements with numerically higher levels will not occur until restart processing of elements with numerically lower levels completes. In other words, delaying an element for CLEANUP_TIMEOUT also delays restart processing of elements with numerically higher is optional.

CLEANUP_TIMEOUT is optional. The default is CLEANUP_TIMEOUT(300).

RESTART_TIMEOUT(time-interval)

Specifies the maximum amount of time, in seconds, that an element is expected to take to issue the IXCARM macro with REQUEST=REGISTER after automatic restart management has restarted it.

If the re-registration does not occur within this interval, automatic restart management issues message IXC803I to inform the operator of the situation and deregisters the element. Any elements that were waiting for this element to become ready will be released from their wait.

time-interval

Must be a decimal number from 1 to 86400 seconds (24 hours).

Note: Many factors can affect how long it will take for a job to restart, so this value should not be set to less than 60 seconds. If *time-interval* is too small, automatic restart management can deregister an element that would have re-registered successfully. When the subsequent re-registration is received, automatic restart management will treat it as an initial registration.

RESTART_TIMEOUT is optional. The default is RESTART_TIMEOUT(300). The IXCARM macro parameter that overrides the default specification is RESTARTTIMEOUT.

READY_TIMEOUT(*time-interval*)

Specifies the maximum amount of time, in seconds, to wait for an element being restarted to issue the IXCARM macro with the REQUEST=READY parameter. If this interval is exceeded, automatic restart management will release any elements that were waiting for this element to become ready. No message is issued for this condition and the element is not deregistered. (If a DISPLAY XCF,ARMSTATUS is done for this element, it will be in the available-to state.)

The value for *time-interval* must be a decimal number from 1 to 86400 seconds (24 hours).

READY_TIMEOUT is optional. The default is READY_TIMEOUT(300).

TERMTYPE(ALLTERM)**TERMTYPE(ELEMTerm)**

Specifies under which conditions MVS should restart this element.

ALLTERM

Indicates that the element should be restarted for all unexpected failures as appropriate. The value specified on an IXCARM REQUEST=REGISTER request for the ELEMBIND keyword determines what types of failures are appropriate.

ELEMTerm

Indicates that the element should be restarted only if the element itself terminates. It should not be restarted if the system terminates.

TERMTYPE is optional. The default is TERMTYPE(ALLTERM). The IXCARM macro parameter that overrides the default specification is TERMTYPE.

When the element is an abstract resource (has a bind to the system on which it is registered), the following applies:

- If the policy specifies TERMTYPE(ELEMTerm), the policy specification for TERMTYPE is ignored. The element will be restarted when the system fails. To suppress all restarts for an element, specify RESTART_ATTEMPTS(0).
- If the policy specifies TERMTYPE(ALLTERM), the policy specification for TERMTYPE is honored. (The element will be restarted when the system fails.)

RESTART_METHOD(*event*,*restart-type*)

Specifies restart text that automatic restart management is to use to restart the element.

Specify one of the following for *event*:

ELEMTerm

Indicates that the persistent restart text is to be overridden by the JCL data set or the command text specified in *restart-type* only when the **element** itself terminates.

SYSTEM

Indicates that the persistent restart text is to be overridden by the JCL data set or the command text specified in *restart-type* only when the **system** the element was running on terminates.

BOTH

Indicates that the persistent restart text is to be overridden by the JCL data set or the command text specified in *restart-type* when either the system the element was running on terminates, or when the element itself terminates.

Specify one of the following for *restart-type*:**PERSIST**

Indicates that MVS is to use the persistent restart text. The persistent restart text is either the JCL or the started task command that previously started the element.

JOB,'jcl-source'

Indicates that MVS is to restart the element as a batch job. '*jcl-source*' is the name of the data set that contains the JCL for restarting the element. This data set name must be enclosed within single quotation marks. The data set must have the same data set characteristics (for instance, LRECL) as standard procedure libraries.

The data set name can:

- Contain symbolic-substitution keywords (such as &SYSCONE)
- Be up to 72 characters in length (after symbolic-substitution keywords are resolved, the data set name cannot exceed 44 characters in length, and the member name cannot exceed 8 characters in length and must be enclosed in parentheses).

Note: Data set names are not validated when a policy is defined except as noted above. They are validated only when a restart is necessary for the element.

STC,'command-text'

Indicates that automatic restart management is to restart this element by issuing the command provided in '*command-text*'. The command text:

- Must be enclosed in single quotation marks
- Can include symbolic substitution keywords
- Length cannot exceed 126 characters
- Can contain a single quotation mark(') but must be represented by specifying two single quotation marks('') in succession.
- Can contain lowercase characters but will be changed to uppercase when the utility program runs. Your command text must not be dependent on lowercase characters.

Note: Command text strings are not validated when a policy is defined except as specified above. They are validated only when a restart is necessary for this element.

Note:

1. If the *jcl-source* or *command-text* will not fit on one line, it can be continued on another line by using single quotation marks to indicate segments. The quotation marks are not counted toward the input length. For example:

```
RESTART_METHOD(BOTH,STC,'S ELEMENT1,PARM1=ABC,'
                     'PARM2=XYZ')
```

is equivalent to specifying:

```
RESTART_METHOD(BOTH,STC,'S ELEMENT1,PARM1=ABC,PARM2=XYZ')
```

2. The RESTART_METHOD can be specified once for ELEMTERM and once for SYSTERM per element. (Specifying RESTART_METHOD more than once for an element when BOTH is specified results in an error.)
3. The RESTART_METHOD can be specified by the element restart exit, IXC_ELEM_RESTART, if one is coded, and by the STARTTXT parameter of the IXCARM macro. If more than one restart method is specified for an element, the order of priority, from highest to lowest, is:
 - a. Element restart exit
 - b. Installation-written policy (if RESTART_METHOD is specified)
 - c. STARTTXT parameter of IXCARM macro
 - d. Policy defaults

4. Blanks specified within quotation marks are significant.

RESTART_METHOD is optional. The default is RESTART_METHOD(BOTH, PERSIST). For started tasks and for elements that represent abstract resources, the IXCARM macro parameter that overrides the default specification is STARTTXT.

To specify the restart text for an element that has a bind to the system on which it is registered (an abstract resource) and restarts only on system termination, use one of the following:

```
RESTART_METHOD(SYSTEM,STC,'command-text')
```

where command-text is a command that causes the element to restart.

```
RESTART_METHOD(SYSTEM,JOB,'jcl-source')
```

where jcl-source is the name of the data set that contains the JCL for restarting the element.

Using wildcard characters

The ELEMENT_NAME and ELEMENT parameters can contain wildcard characters. The wildcard characters are:

?

A question mark matches one and only one character, and can be specified more than once anywhere in the name.

An asterisk matches a series of characters, and can be specified only at the end of a name.

These characters allow a single set of element-related parameters in a policy to apply to multiple elements with similar names and needs.

If two or more element names with wildcard characters in a policy match the name of the element being restarted, the most specific takes priority. For example, if MVS is restarting ELEM1, and the policy contains elements with the names: ELEM?, ELEM*, and ELEM1*, the information for ELEM1* will be used.

Examples of matching

If the element name is ABC?, this would match:

```
ABCD
```

but not:

```
ABCDE or ABC
```

If the element name is AB?D*, this would match

```
ABCDEFGHIK
```

and:

```
ABCD and ABCDE and ABYDEFGHI
```

Using system symbol substitution

Automatic restart management supports both system symbols that the system provides and static system symbols that your installation defines for:

- *jcl-source* on the RESTART_METHOD parameter
- *command-text* on the RESTART_METHOD parameter

When symbols are used with RESTART_METHOD, note that the symbols are resolved using the symbol table from the system where the job **initially registered** as an element of automatic restart management.

The symbols that were in effect on that system when the job initially registered are used; symbols added or changed subsequently are not used.

See the topic on sharing parmlib members in *z/OS MVS Initialization and Tuning Reference* for lists of system symbols that automatic restart management supports.

CFRM parameters for administrative data utility

The CFRM policy information describes the coupling facilities (CF) and structures (STRUCTURE) that can be used in the sysplex once an administrative policy is activated.

The syntax of the CFRM parameters for the Administrative Data Utility is shown in [Figure 73 on page 346](#)

```
DEFINE POLICY NAME(polname) REPLACE(NO | YES | timestring)
CF NAME(cfname)
  TYPE(tttttt)
  MFG(mmm)
  PLANT(pp)
  SEQUENCE(nnnnnnnnnnnn)
  PARTITION(h | hh)
  [SIDE(0 | 1)]
  [CPCID(nn)]
  [DUMPSPACE(size[u])]
  [SITE(SITE1 | SITE2)]
STRUCTURE NAME(strname)
  SIZE(size[u])
  [INITSIZE(itsize[u])]
  [MINSIZE(minsize[u])]
  [SCMMAXSIZE(scmmassize[u])]
  [SCMALGORITHM(algorithm)]
  [ALLOWAUTOALT(NO | YES)]
  [FULLTHRESHOLD(value)]
  [PREFLIST(cfname1,cfname2,...,cfname8)]
  [EXCLLIST(strname1,strname2,...,strname8)]
  [REBUILDPERCENT(value)]
  [DUPLEX(DISABLED | ALLOWED[,dupopts] | ENABLED[,dupopts])]
  [RECPRTY(value)]
  [SUBNOTIFYDELAY(delaytime)]
  [LISTNOTIFYDELAY(listnotifydelay)]
  [KEYRNOTIFYDELAY(keyrnotifydelay)]
  [ENFORCEORDER(NO | YES)]
  [ALLOWREALLOCATE(YES | NO)]
  [ENCRYPT(NO | YES)]
```

Figure 73. Syntax of CFRM parameters for the Administrative Data Utility

The SETXCF START,POLICY,TYPE=CFRM operator command is used to activate an administrative policy. If no policy is active for CFRM, then the specified administrative policy is activated immediately. Otherwise, a transition to the newly activated policy is required. For an allocated structure, some subparameters can take effect immediately. STRUCTURE subparameters that take effect immediately include:

- ALLOWAUTOALT
- FULLTHRESHOLD
- REBUILDPERCENT
- DUPLEX (see the DUPLEX parameter for special conditions)
- ALLOWREALLOCATE
- RECPRTY
- SUBNOTIFYDELAY
- LISTNOTIFYDELAY
- KEYRNOTIFYDELAY

When running under z/VM®, see [“Setting up a guest coupling environment under z/VM” on page 420](#) for information about values that you must specify in the CFRM policy.

DEFINE POLICY NAME(polname)

Specifies the beginning of the definition of a new administrative policy. The limit for the number of policies that can be defined in a couple data set is established when the couple data set is formatted.

polname specifies the name of the policy. The name can be from 1 - 8 characters in length. Valid characters are A-Z and 0-9. The policy name must start with an alphabetic character.

A policy name can be defined only once in a single run of the administrative data utility.

See “Coding the administrative data utility” on page 336 for other utility control statements; included here are only the portion that applies only to CFRM.

REPLACE(NO)**REPLACE(YES)****REPLACE(timestring)**

See the REPLACE parameter under “Coding the administrative data utility” on page 336.

CF

Specifies the definition of a coupling facility within the scope of the named policy. The location of the information that is needed to define the coupling facility in a CFRM policy depends on the processor on which the coupling facility is installed.

The limit for the number of coupling facilities that can be defined in a policy is established when the CFRM couple data set is formatted.

The coupling facility must appear in the preference list of at least one coupling facility structure that is defined within the policy or the administrative policy utility flags the statement as an error.

NAME(cfname)

Specifies the 1 - 8 character length name of the coupling facility. The valid characters are uppercase alphabetic characters (A-Z), numeric characters (0-9), national characters (\$,@,#), or an underscore (_). The **cfname** must start with an alphabetic character (A-Z).

Consider defining your coupling facility name to match the name of the LPAR in which the coupling facility is to run.

NAME is a required parameter.

The following set of parameters describe the unique coupling facility that is defined in the policy.

TYPE(tttttt)

Specifies the 6-character machine type. The valid characters are uppercase alphabetic characters (A-Z) and numeric characters (0-9), padded with leading zeros if necessary. TYPE is a required parameter.

MFG(mmm)

Specifies the 3-character manufacturer identification. The valid characters are uppercase alphabetic characters (A-Z) and numeric characters (0-9). MFG is a required parameter.

PLANT(pp)

Specifies the 2-character plant of manufacture code. The valid characters are uppercase alphabetic characters (A-Z) and numeric characters (0-9). PLANT is a required parameter.

SEQUENCE(nnnnnnnnnnnn)

Specifies the 12-character serial number. The valid characters are uppercase alphabetic characters (A-Z) and numeric characters (0-9), padded with leading zeros if necessary. SEQUENCE is a required parameter.

PARTITION(h) or (hh)

Specifies the 1- or 2-hexadecimal digit qualifier to uniquely identify and associate a coupling facility to a specific PR/SM partition.

- If the coupling facility is defined on a System z9®, z990, and z890 processor, then the PARTITION value must match the partition ID associated on the activation profile for the CF image on the support element or hardware master console.

- If the coupling facility is defined on a pre-z990 or pre-z890 or earlier processor, then the PARTITION value must match the partition number specified by HCD/IOCP in the IOCDs for that partition and can be in the range of 01-0F.
- Otherwise, the PARTITION value can be in the range of 0-F and is the same as the partition number defined in HCD.

See [“Configuring a processor and a coupling facility” on page 42](#) for information about specifying the partition identifier.

PARTITION is a required parameter.

SIDE(0)

SIDE(1)

Specifies the 1-numeric character to identify the coupling facility to a specific physical side of a CPC running in Physically Partitioned mode. You should not split or merge physical sides on which a coupling facility resides.

Use caution when splitting or merging physical sides of a processor that contains a coupling facility.

SIDE is an optional parameter. Its omission implies that the coupling facility resides on a single image CPC.

- SIDE(0) — Side 0 of a physically partitioned machine. Do not specify for a Single Image CPC.
- SIDE(1) — Side 1 of a physically partitioned machine. Do not specify for a Single Image CPC.

CPCID(nn)

Specifies the 2-hexadecimal digit qualifier (00-FF) to identify a coupling facility to a specific CPC in a CMOS processor. CPCID is an optional parameter.

DUMPSPACE(size[u])

Specifies the amount of space to be reserved in the coupling facility for dumping structures allocated in the coupling facility. The number is specified in an integer unit of *u*, where *u* is specified as K (kilobytes), M (megabytes), G (gigabytes) or T (terabytes). K is the default unit when no size unit is specified. The number can be 1 - 9 decimal digits long. The largest size that can be specified is 1T. Sizes larger than 1T causes message IXC730I to be displayed indicating that the number specified is not within the allowable range for the keyword value.

DUMPSPACE is an optional parameter. If omitted, no storage in the coupling facility is reserved for dumping.

For information about specifying the amount of dump space, see [“Determining the amount of coupling facility dump space” on page 65](#).

SITE(SITE1)

SITE(SITE2)

Specifies the site at which the coupling facility resides. When specified, this information (along with status information from a Recovery Manager (for example, GDPS)), will be used in break duplexing decisions to keep the structure instance at the recovery site. SITE changes take effect when the policy is activated.

Note: With APAR OA31601, the recovery site provided to XCF through IXCCFCM is ignored. The recovery site does not affect which structure is kept when coupling facility structure duplexing is stopped. IXCQUERY or the DISPLAY XCF command does not provide the recovery site. Because using the recovery site might cause you to lose duplexed coupling facility structure data in the event of a coupling facility failure at the recovery site, disabling use of the recovery site with APAR OA31601 helps you avoid the problem.

This keyword should be considered when specifying the DUPLEX **dupsite** and **dupmode** parameters as part of the STRUCTURE definition.

STRUCTURE

Specifies the definition of a structure within the scope of the named policy. The limit for the number of structures that can be defined in a policy is established when the CFRM couple data set is formatted.

NAME(strname)

Specifies the 1-16 character name of the structure. The valid characters are numeric characters, uppercase alphabetic characters, national characters (\$,@,#), or an underscore (_). The *strname* must start with an alphabetic character (A-Z). IBM names begin with SYS, or the letters A through I.

Structures that are used for XCF signaling must begin with the letters IXC.

NAME is a required parameter.

SIZE(size[u])

Specifies the maximum amount of space to be allocated for the structure in the coupling facility. The number is specified in an integer unit of *u*, where *u* is specified as K (kilobytes), M (megabytes), G (gigabytes) or T (terabytes). K is the default unit when no size unit is specified. The number can be 1 - 9 decimal digits long.

The maximum structure size is the largest to which the structure can be altered. If a structure size larger than the maximum is required, you must modify the CFRM policy to specify the larger size. When you start the modified policy, the system can reallocate the new structure, which likely will be disruptive to your operation.

Specifying too large a maximum structure size can waste coupling facility resources. Especially when the SIZE is significantly larger than INITSIZE, it might even cause the structure to become unallocatable, depending on the coupling facility and its CFLEVEL.

SIZE is a required parameter. The largest size that can be specified is 1T. Sizes larger than 1T causes message IXC730I to be displayed indicating that the number specified is not within the allowable range for the keyword value.

INITSIZE(initsize[u])

Specifies the initial amount of space to be allocated for the structure in the coupling facility. The number is specified in an integer unit of *u*, where *u* is specified as K (kilobytes), M (megabytes), G (gigabytes) or T (terabytes). K is the default unit when no size unit is specified. The number must be 1 - 9 decimal digits long. The INITSIZE value must be less than or equal to the SIZE value. Otherwise, the system issues error message IXC745I.

INITSIZE is an optional parameter. If not specified, the system uses the SIZE parameter. The largest size that can be specified is 1T. Sizes larger than 1T causes message IXC730I to be displayed indicating that the number specified is not within the allowable range for the keyword value.

MINSIZE(minsize[u])

Specifies the smallest size to which the structure can ever be altered, preventing allocation of an unusable structure. The number is specified in an integer unit of *u*, where *u* is specified as K (kilobytes), M (megabytes), G (gigabytes) or T (terabytes). K is the default unit when no size unit is specified.

Structure alter can be initiated by operator command, programming interface, or automatic alter (see ALLOWAUTOALT). MINSIZE will also serve as a minimum bound for the structure size on all structure allocation requests that occur on systems at OS/390 Release 10 and above (including connect and rebuild connect) with an exception that system-managed rebuild can allocate the rebuild new structure with a size smaller than the MINSIZE value. This can occur when there is not enough space in any of the coupling facilities in the preference list to satisfy the MINSIZE requirement for the allocation of the rebuild new structure, but there is enough space in at least one of the coupling facilities in the preference list to allocate the rebuild new structure with a size that allows inuse objects to be copied from the rebuild old structure. In this case, instead of stopping the system-managed rebuild because the MINSIZE requirement could not be satisfied, the system allows the rebuild to continue.

The number can be 1 - 9 decimal digits long. The MINSIZE value must be less than or equal to the INITSIZE value, or less than or equal to the SIZE value if INITSIZE is not specified.

MINSIZE is an optional parameter. The largest size that can be specified is 1T. Sizes larger than 1T will cause message IXC730I to be displayed indicating that the number specified is not within

the allowable range for the keyword value. If not specified and ALLOWAUTOALT(NO) is specified or defaulted to, the MINSIZE used will be 0. If not specified and ALLOWAUTOALT(YES) is specified, the system will set MINSIZE to 75% of INITSIZE, or to 75% of SIZE if INITSIZE is not specified.

SCMMAXSIZE(scmmaxsize[*u*])

Specifies the maximum amount of storage-class ("flash") memory that is assignable for use by this structure in the coupling facility. The number is specified in an integer unit of *u*, where *u* is specified as K (kilobytes), M (megabytes), G (gigabytes), or T (terabytes). K is the default unit when no size unit is specified. Valid values for the number are 1 - 9 decimal digits long and greater than 0 if this keyword is specified.

Storage-class memory is not allocated to the structure until required for use. It is therefore possible to overcommit the storage-class memory that is configured to the coupling facility. It is possible to define the CFRM policy such that the sum of the SCMMAXSIZE values for allocated structures exceeds the total amount of storage-class memory that is available to the coupling facility.

Note:

1. Do not overcommit storage-class memory. This ensures that you have the desired amount of storage available in the event of an application failure that causes the structure to fill up.
2. Coupling facilities defined on the z17 processor or later (CFLEVEL 26 and above) no longer support storage-class memory. Refer to application-specific documentation for alternative means of providing structure overflow capacity

SCMMAXSIZE is an optional parameter. If not specified, the coupling facility does not spill excess structure objects into storage-class memory for this structure. The following values are the largest values that you can specify:

- 15T
- 16383G
- 16777215M
- 999999999K or 999999999

For information on the use of storage-class memory to reduce the probability of structure-full conditions, see [“Requesting the use of storage-class memory” on page 54](#).

SCMALGORITHM(algorithm)

Identifies the algorithm that the coupling facility uses to control the movement of structure objects between coupling facility real storage and storage-class memory in an effort to mitigate the performance penalty associated with its use.

The valid values of SCMALGORITHM and the structures that they support are described in the following table.

Note: Specification of SCMMAXSIZE and the related keywords for inappropriate structure types might have unpredictable results, potentially including connect failures or performance degradation.

Table 30. SCMALGORITHM value descriptions

SCMALGORITHM Value	Description	Structure Types
KEYPRIORITY1	<p>The high-order byte of the list entry key is treated as a migration priority indicator for lists 1 - 512. Valid values for the priority byte are:</p> <ul style="list-style-type: none"> '00'x - '09'x, where lower values indicate higher priority 'F4' - 'F6'x are application-specific values 	IBM MQ application structure

SCMALGORITHM is a required parameter when SCMMAXSIZE is specified and is not allowed for specification otherwise.

ALLOWAUTOALT(NO)**ALLOWAUTOALT(YES)**

Specifies the installation's request to allow system-initiated alters (automatic alter) for this structure. For structure alter processing to be started for a structure, alter must be permitted (see the SETXCF MODIFY command) and the exploiter must also allow alter. The ALLOWAUTOALT specification affects the default value for MINSIZE.

See FULLTHRESHOLD and MINSIZE for related information.

ALLOWAUTOALT takes effect immediately with policy activation.

ALLOWAUTOALT is an optional parameter. NO is the default, when ALLOWAUTOALT is not specified.

FULLTHRESHOLD(value)

Specifies a percentage value used by the system to control structure full monitoring and automatic alter (see ALLOWAUTOALT). The value specifies a percent full threshold for the structure. For a cache structure, the percent full is based on changed coupling facility structure objects. For a list or lock structure, the percent full is based on in use coupling facility structure objects. This number is specified as a percentage and can be 1 - 3 decimal digits long (0-100). The number must be greater than or equal to 0 and less than or equal to 100.

Specifying 0 will result in no structure full monitoring or automatic alter processing for the structure. If a value other than 0 is specified or defaulted to for FULLTHRESHOLD, then the value will be used as a percent full threshold for structure full monitoring.

If the exploiter allows alter, ALLOWAUTOALT(YES) is specified, starting alter processing for the structure is permitted, and the FULLTHRESHOLD value is reached, automatic alter processing can be initiated. When the FULLTHRESHOLD value is reached, structure full monitoring will alert the installation through an IXC585E message and automatically start structure alter processing to relieve the storage shortage for the object whose storage is in short supply. Before starting the structure alter, automatic alter will issue message IXC588I to the system log to externalize the alter request.

If the exploiter does not allow alter, or starting alter processing for the structure is not permitted, the ALLOWAUTOALT specification is ignored and a nonzero value for FULLTHRESHOLD is only used by structure full monitoring to alert the installation.

FULLTHRESHOLD takes effect immediately with policy activation.

FULLTHRESHOLD is an optional parameter. 80% is the default, when FULLTHRESHOLD is not specified.

PREFLIST(cfname1,cfname2,...,cfname8)

Specifies 1-8 coupling facility names from which the system is to choose when allocating a structure in a coupling facility. If ENFORCEORDER(NO) is specified, the system attempts to allocate the structure in the first coupling facility in the preference list that meets the following allocation criteria regulated by the duplex site preference, and listed in order of relative importance from most important to least important as described in "How MVS uses the lists" in ["Identifying the coupling facility structures" on page 49](#).

The system assumes certain criteria when selecting a coupling facility for new structure allocation in structure duplexing. The system always assumes LOCATION=OTHER when selecting the coupling facility. As mentioned previously, the coupling facility chosen by the system will, if possible, be failure-independent with respect to the coupling facility containing the old structure. Lastly, if the level of connectivity to the new structure is less than that to the old structure, the action taken by the system is LESSCONNECTION=TERMINATE.

In system-managed duplexing rebuild, the system also requires that the coupling facility for the new structure allocation have CF-to-CF connectivity to the coupling facility containing the structure to be duplexed. See ["Selecting a coupling facility for the duplexed pair of structures" on page 75](#) for a description of the criteria used by the system when allocating a duplexed structure in a coupling facility.

If there is no coupling facility in the preference list that meets all these criteria, then the system determines the coupling facility that most closely meets the criteria. See ["How MVS uses the lists" on page 58](#) for information about how the system chooses the coupling facility in the preference list that most closely meets the requirements of the connect request.

If the structure you are defining can only reside in a coupling facility with a certain attribute (such as CFLEVEL or nonvolatility), be sure to include only those coupling facilities in the preference list for the structure. Two or more coupling facilities are required when you specify DUPLEX(ALLOWED) or DUPLEX(ENABLED).

If ENFORCEORDER(YES) is specified, the system will not reorder the coupling facilities in the preference list, but will attempt to allocate the structure in the exact order specified in the preference list, regulated by the SAMESITEONLY duplex site preference. Therefore, when constructing your preference list, be sure to place the coupling facilities with the preferred attributes before any coupling facilities without the desired attributes when ENFORCEORDER(YES) is specified.

Refer to "Developing preference and exclusion lists" in "Understanding Preference and exclusion lists" in ["Identifying the coupling facility structures" on page 49](#) for additional considerations.

ENFORCEORDER(YES) is mutually exclusive with EXCLLIST.

PREFLIST is a required parameter.

EXCLLIST(strname1,strname2,...,strname8)

Specifies the list of 1 to 8 coupling facility structure names with which this structure should not share the same coupling facility. The system attempts to honor the exclusion request, but will not fail a request to allocate a structure when all other requirements for structure allocation have been satisfied and the exclusion list cannot be honored. However, if all other attributes are equal, a coupling facility containing only one instance of a duplexed structure from the exclusion list is selected over a coupling facility containing a simplex structure from the exclusion list.

EXCLLIST is mutually exclusive with ENFORCEORDER(YES).

EXCLLIST is an optional parameter.

REBUILDPERCENT(value)

Specifies, as a percent of lost connectivity to a structure, when MVS is to initiate a user-managed rebuild. Use of REBUILDPERCENT requires that all active connections to the structure support user-managed rebuild and that the structure is not being used for XCF signaling.

This number is specified as a nonzero percentage and can be 1 - 3 decimal digits long (1-100).

REBUILDPERCENT takes effect immediately with policy activation.

REBUILDPERCENT is an optional parameter. When REBUILDPERCENT is not specified, the default value is 1%.

See [“Specifying a rebuild threshold”](#) on page 59 for information about MVS-initiated rebuild.

DUPLEX(DISABLED)

DUPLEX(ALLOWED[,dupopts])

DUPLEX(ENABLED[,dupopts])

Specifies the installation's request for duplexing rebuild of the structure.

DISABLED

Neither user-managed nor system-managed duplexing rebuild can be started for the specified structure. If a duplexing rebuild is in progress, it is stopped.

ALLOWED[,dupopts]

The application can initiate its own user-managed or system-managed duplexing rebuild or an operator can start either method of duplexing rebuild with the SETXCF START,REBUILD,DUPLEX command. However, with the exception of duplexing being stopped for the structure and then duplexed again within a REALLOCATE process, the system does not make any attempts to maintain the duplexed status of the structure. The application or operator must initiate another duplexing operation if the current duplexing is stopped.

ENABLED[,dupopts]

The system initiates and attempts to maintain a user-managed or system-managed duplexing rebuild for the structure. When a duplexing rebuild process is stopped, the system attempts to initiate duplexing again. If the operator stops the duplexing operation, the system ensures that, for the system's first attempt to duplex the structure again, the coupling facility most recently used to contain the previous instance of the duplexed structure is not used again to contain the structure. However, for any attempts thereafter to duplex the structure again, the system might choose the same coupling facility for allocation that the structure just vacated.

Coupling facility maintenance might require you to move an instance of a duplexed structure from a coupling facility. In order to do this, see [“Guidelines for coupling facility reconfiguration”](#) on page 427 for more information.

dupopts

One or more of the following, each separated by a comma:

```
dupsite
dupmode | (dupmode) | (dupmode1,dupmode2)
```

dupsite

The **dupsite** parameter is used to determine how a CF SITE is used when determining CF importance and eligibility for duplexed CF structure allocation. All systems in the sysplex must be at z/OS V2R2 or higher for this parameter to be fully effective. Because lower-level systems in the sysplex are unaware of this parameter, any CF structure allocation decisions performed by them are processed as if the **dupsite** parameter were not specified.

Specify the CROSSSITE, SAMESITE, or SAMESITEONLY keywords to influence CF structure allocation decisions. The **dupsite** parameter can be specified as one of the following values:

ANYSITE

CF SITE specification is not used when determining CF importance and eligibility for duplexed CF structure allocation. This is the default when the **dupsite** parameter is not specified.

CROSSSITE

It is preferred that the duplexed structure instances be allocated across sites according to the CF SITE specification.

SAMESITE

It is preferred that the duplexed structure instances be allocated in the same site according to the CF SITE specification.

SAMESITEONLY

It is required that the duplexed structure instances be allocated in the same site according to the CF SITE specification.

dupmode|dupmode1|dupmode2

This parameter is used to determine whether synchronous or asynchronous duplexing should be used. This parameter can only be used when the couple data set is formatted with ASYNCDUPLEX. All options can be used for lock structures. Only the default option can be used for other structure types.

This parameter can be used in a pair of parameters that determine whether synchronous or asynchronous duplexing should be used. When used as a pair, the more synchronous option applies to same-site duplexing. The other option is used for other configurations. For example, if one of the options is SYNCONLY, same-site duplexing is done synchronously. If one of the options is ASYNCONLY, same-site duplexing uses the other option.

SYNCONLY

The structure is duplexed by using synchronous duplexing only. Either user-managed duplexing or system-managed synchronous duplexing is used. This is the default when the **dupmode** parameter is not specified.

ASYNCONLY

The structure is duplexed by using asynchronous duplexing only. System-managed asynchronous duplexing is used.

When this keyword is used, system-managed duplexing rebuild allocation for the structure requires a CFLEVEL that supports asynchronous duplexing (CFLEVEL>=21).

ASYNCON

The structure is duplexed by using asynchronous duplexing, if possible. If not possible, the structure is duplexed by using synchronous duplexing. Asynchronous duplexing might not be possible due to CFLEVEL or due to lack of connector or application support.

DUPLEX(DISABLED) and the default dupmode of SYNCONLY both indicate that a structure will be allocated without the ability to participate in asynchronous duplexing. Other uses of dupmode indicate that a structure should be allocated with the ability to participate in asynchronous duplexing.

Otherwise, a change to the **DUPLEX** parameter takes effect immediately with policy activation.

Examples: You might want to consider asynchronous duplexing for a structure that uses DUPLEX(DISABLED). You might have not done duplexing because the “mainline” costs were too high. You might want to introduce asynchronous duplexing given that it does not have the “mainline” overhead of synchronous duplexing, and the benefits of asynchronous duplexing outweigh the costs of not duplexing. The structure would be changed to DUPLEX(ENABLED,ASYNCONLY).

You might want to consider asynchronous duplexing for a structure that uses DUPLEX(ENABLED). You would like to migrate to asynchronous duplexing to eliminate the “mainline” overhead of synchronous duplexing. The structure would be changed to DUPLEX(ENABLED,ASYNCON).

You might want to consider asynchronous duplexing for a structure that uses DUPLEX(ENABLED,SAMESITEONLY). For SAMESITE duplexing, you might not want to introduce the potential duplexing failover penalty that is associated with asynchronous duplexing. But you might want to introduce CROSSSITE asynchronous duplexing given that it does not have the “mainline” overhead of synchronous duplexing, and the benefits of asynchronous duplexing outweigh the costs of NOT duplexing. The structure could be changed to DUPLEX(ENABLED,(SYNCONLY,ASYNCONLY)).

You might want to consider asynchronous duplexing for a structure that uses DUPLEX(ENABLED,SAMESITE). For SAMESITE duplexing, you might not want to introduce the potential duplexing failover penalty that is associated with asynchronous duplexing. But you might want to eliminate the “mainline” overhead of asynchronous duplexing when duplexed CROSSSITE. The structure would be changed to DUPLEX(ENABLED,(SYNCONLY,ASYNCON)).

Table 31. DUPLEX keywords. This table shows the relationships between duplexing mode and the DUPLEX keyword.

DUPLEX keywords	Duplexing mode	Same-site duplexing	Cross-site duplexing
DISABLED	-	-	-
+SAMEITEONLY	,ASYNCONLY	Async	-
+SAMEITEONLY	,ASYNC	Async/Sync	-
+SAMEITEONLY	[,SYNCONLY]	Sync	-
*other	,ASYNCONLY	Async	Async
*other	,(ASYNCONLY,ASYNCONLY)	Async/Sync	Async
*other	,(SYNCONLY,ASYNCONLY)	Sync	Async
*other	,ASYNC	Async/Sync	Async/Sync
*other	,(SYNCONLY,ASYNCONLY)	Sync	Async/Sync
*other	[,SYNCONLY]	Sync	Sync
+ = {ALLOWED ENABLED}			
*other = {ALLOWED ENABLED} [,SAMEITE ANYSITE CROSSITE]			

DUPLEX is an optional parameter. DISABLED is the default, when DUPLEX is not specified.

SUBNOTIFYDELAY (delaytime)

Specifies the number of microseconds for sublist notification delay time (SLND time). This value refers to delay between the time when a single selected shared message queue exploiter instance is notified of sublist transition from the empty to not-empty state and the time when the other instances are notified. It can be that the other instances are never notified depending on the processing done by the initial exploiter. See *z/OS MVS Programming: Sysplex Services Guide* for an explanation of the sublist monitoring function.

The value that is specified for SUBNOTIFYDELAY can be 1 to 7 decimal digits in a range of 0 to 1000000 (1 million) microseconds.

SUBNOTIFYDELAY only applies to structures meeting certain criteria and is ignored for all other structures. The structure criteria for the SUBNOTIFYDELAY parameter are:

- The structure must be allocated as a keyed list structure
- The structure must have EMCs allocated
- The structure must be allocated in a CFLEVEL 16 or higher

SUBNOTIFYDELAY takes effect immediately with policy activation.

The SUBNOTIFYDELAY value cannot exceed the CF model-dependent Notification Delay Limit (NDL) value that is determined by the CF where the structure is allocated. Therefore, the SUBNOTIFYDELAY value that takes effect is the smaller of the SUBNOTIFYDELAY value and the CF-dependent NDL value.

SUBNOTIFYDELAY is an optional parameter. The default value when SUBNOTIFYDELAY is not specified is 5000 microseconds.

LISTNOTIFYDELAY(listnotifydelay)

Specifies the number of microseconds for the list notification delay time (LND time). This value refers to the delay between the time when a single registered list monitor is notified of a list state transition from the empty to not-empty state and the time when the other registered list monitor instances of a list are notified. It can be that the other instances are never notified, depending on the processing done by the initial notified monitor. See *z/OS MVS Programming: Sysplex Services Guide* for an explanation of the list monitoring function.

The value that is specified for LISTNOTIFYDELAY can be 1 to 7 decimal digits, in a range of 0 to 1000000 (1 million) microseconds. LISTNOTIFYDELAY only applies to structure meeting certain criteria and is ignored for all other structures. The structure criteria for the LISTNOTIFYDELAY parameter is as follows:

- The structure must be allocated as a list structure.
- The structure must be allocated in a CFLEVEL=22 or higher.

LISTNOTIFYDELAY takes effect immediately with policy activation. The LISTNOTIFYDELAY value cannot exceed the CF model dependent NDL (Notification Delay Limit) value that is determined by the CF where the structure is allocated. Therefore, the LISTNOTIFYDELAY value that takes effect, is the smaller of the LISTNOTIFYDELAY value and the CF dependent NDL value.

LISTNOTIFYDELAY is an optional parameter. The default value when LISTNOTIFYDELAY is not specified is 0 microseconds.

KEYRNOTIFYDELAY(KEYRNOTIFYDELAY)

Specifies the number of microseconds for the keyrange notification delay time (KRND time). This value refers to the delay between the time when a single registered keyrange monitor is notified of a keyrange state transition from the empty to not-empty state and the time when the other registered keyrange monitoring instances of a list keyrange are notified. It can be that the other instances are never notified, depending on the processing done by the initial notified monitor. See [*z/OS MVS Programming: Sysplex Services Guide*](#) for an explanation of the list monitoring by keyrange function.

The value that is specified for KEYRNOTIFYDELAY can be 1 to 7 decimal digits, in a range of 0 to 1000000 (1 million) microseconds. KEYRNOTIFYDELAY only applies to structure meeting certain criteria and is ignored for all other structures. The structure criteria for the KEYRNOTIFYDELAY parameter is as follows:

- The structure must be allocated as a keyed list structure.
- The structure must be allocated in a CFLEVEL=22 or higher.

KEYRNOTIFYDELAY takes effect immediately with policy activation. The KEYRNOTIFYDELAY value cannot exceed the CF model dependent NDL (Notification Delay Limit) value that is determined by the CF where the structure is allocated. Therefore, the KEYRNOTIFYDELAY value that takes effect, is the smaller of the KEYRNOTIFYDELAY value and the CF dependent NDL value.

KEYRNOTIFYDELAY is an optional parameter. The default value when KEYRNOTIFYDELAY is not specified is 0 microseconds.

ENFORCEORDER(NO)**ENFORCEORDER(YES)**

When ENFORCEORDER(YES) is specified, allocation attempts adhere to the PREFLIST more closely, giving it a higher priority than such factors as system weight connectivity and optimum size. Use of this option can help ensure failure-isolation between a coupling facility and the systems that utilize it (for example, when these resources reside on the same CEC).

ENFORCEORDER is an optional parameter. ENFORCEORDER(NO) is the default, when ENFORCEORDER is not specified.

RECPRTY(value)

This statement specifies the priority to be given to the structure for LOSSCONN recovery (system loses connectivity to a coupling facility) and policy-initiated duplexing for DUPLEX(ENABLED) structures. The system might defer processing for a "less important" (higher numeric priority value) structure to prevent the "less important" recovery processing from interfering with recovery processing for a "more important" (lower numeric priority value) structure. Note that LOSSCONN recovery for a "more important" structure might be impacted when that recovery depends on processing for a "less important" structure. For this reason, be extremely careful when assigning a RECPRTY value.

This number is specified as a 1-digit decimal value (1-4).

RECPRTY does not apply to XCF signaling structures.

RECPRTY takes effect immediately with policy activation.

RECPRTY is an optional parameter. When RECPRTY is not specified, the system takes a default. When the structure name is ISGLOCK, the default is 1. Otherwise, the default is 3.

ALLOWREALLOCATE(YES)

ALLOWREALLOCATE(NO)

Specifies the installation's request for REALLOCATE processing when evaluation of the allocated structure determines that the structure needs rebuild processing. The REALLOCATE process recognizes the need for rebuild processing when:

- There is a change to the policy definition for the structure affecting the structure size or location
- The structure is not optimally located

The ALLOWREALLOCATE option provides the installation with a structure-level control to preclude REALLOCATE processing from targeting structures for relocation or policy change activation. This can be useful for structures for which rebuild processing is disruptive based on workload or usage. As one example, IMS batch (DLI batch) jobs are terminated with U3303 abends when a structure rebuild is initiated for certain IMS structures. To prevent the REALLOCATE process from targeting these IMS structures for relocation or policy change activation, the installation can activate an administrative policy with ALLOWREALLOCATE(NO) specified for these specific IMS structures.

By evaluating an allocated structure, the REALLOCATE process can recognize that the structure is optimally located and immediately complete a change to the policy definition for the structure when the change does not affect the size. Specifying ALLOWREALLOCATE(NO) does not preclude this capability of the REALLOCATE process, because completing the pending policy change does not require relocation of the structure instances.

In addition, REALLOCATE processing is a triggering event for MVS-initiated duplexing rebuild when DUPLEX(ENABLED) is specified for the structure in the CFRM policy and the structure is not duplexed. Specifying ALLOWREALLOCATE(NO) does not prevent MVS-initiated duplexing rebuild, because by specifying DUPLEX(ENABLED) the installation is requesting MVS to attempt to maintain the duplexed status of the structure.

See the SETXCF START/STOP command in *z/OS MVS System Commands* for a description of the REALLOCATE process. See [“Using REALLOCATE or POPULATECF to relocate coupling facility structures” on page 90](#) for information about using REALLOCATE or POPULATECF to relocate coupling facility structures.

YES

The REALLOCATE process evaluates the allocated structure. When a policy change for size or location is pending or structure instances are not optimally located, the structure is selected as the target of the REALLOCATE process. The REALLOCATE process uses structure rebuild processing to make the adjustments.

NO

The REALLOCATE process evaluates the allocated structure but does not select the structure as the target of the REALLOCATE process for relocation or policy change activation. When NO is specified, it is still possible for the REALLOCATE process to take the following actions if they are applicable:

- Complete a pending policy update when the pending change does not affect the size or location
- Trigger an MVS-initiated duplexing rebuild when DUPLEX(ENABLED) is specified for the structure and the structure is not duplexed

Changes to the ALLOWREALLOCATE specification take effect immediately with policy activation.

ALLOWREALLOCATE is an optional parameter. When ALLOWREALLOCATE is not specified, the default value is YES.

ENCRYPT(YES)

ENCRYPT(NO)

Specifies whether list and cache structure entry data and entry adjunct data written to the structure and residing in the structure should be encrypted. The structure entry and entry adjunct data is in an encrypted format while the data is being transferred to and from the coupling facility and while

the data resides in the coupling facility structure. Encrypted data is decrypted when read from the structure.

XCF uses Advanced Encryption Standard (AES) 256-bit keys encrypted under an AES Master Key to encrypt and decrypt structure data.

See [“Encrypting coupling facility structure data”](#) on page 80 for information on system requirements that must be met prior to using the ENCRYPT parameter on a structure definition and allocating encrypted structures.

In addition to authorizing the use of the utility (see [“Authorizing use of the utility”](#) on page 335) for updating CFRM policies and CFRM couple data sets, you must ensure that users of the utility must have at least READ access to defined ICSF CSFSERV CSFKGN and CSFSERV CSFKYT resource profiles when ENCRYPT(YES) is specified on a structure definition.

Note that all systems in the sysplex must be at z/OS V2R3 or higher for this parameter to be fully effective. IBM recommends that before using the ENCRYPT parameter, make sure that all systems in the sysplex are running z/OS V2R3 or higher.

Systems running on a lower level of z/OS (V2R2 or below) in the sysplex are unable to encrypt and decrypt structure data. Connection requests from a lower level system to an encrypted structure are not allowed.

Note: For report and for the purpose of messages, the size unit that is specified on DUMPSPACE, SIZE, INITSIZE, and MINSIZE can be converted to the largest size unit that can be used to represent the size and avoids any rounding. For example, specifying INITSIZE(1048576K) might cause INITSIZE to be converted to INITSIZE(1G) in messages. Specifying INITSIZE(120000K) will not cause INITSIZE to be converted because it is not an even multiple of megabytes, gigabytes, or terabytes.

See [“Requesting structure size”](#) on page 50 for more information about requesting the size for structures. See [“Allowing a structure to be altered automatically”](#) on page 63 for more information about system-initiated alter processing.

LOGR keywords and parameters for the administrative data utility

The sysplex scope LOGR policy information describes the characteristics of a log stream and a coupling facility structure that will be used when there are active connections to the log stream. You can specify the REPORT keyword for any of the requests.

The sysplex scope LOGR couple data set must be activated and available before you add log stream and structure definitions to it. See [“Format the sysplex scope LOGR couple data set and make it available to the sysplex”](#) on page 276.

When you intend to specify LOGR policy information, you will need to execute the IXCMIAPU utility on a system in the sysplex that is not using either of the single-system scope LOGRY or LOGRZ policies. The resources in the sysplex scope LOGR policy are managed as separate resources from the log stream resources defined in either of the single-system scope LOGRY or LOGRZ policies.

Note: When using a system logger single-system scope LOGRY or LOGRZ policy within the sysplex, see [“LOGRY and LOGRZ keywords and parameters for the administrative data utility”](#) on page 386.

For guidance on planning your LOGR couple data set parameters, see [Chapter 10, “Planning for system logger applications,”](#) on page 213.

For specific information about defining RRS log streams, see [z/OS MVS Programming: Resource Recovery](#).

You can also use the IXGINVNT macro to define the LOGR policy information. Note that the parameters of the administrative data utility are not identical to the IXGINVNT parameters for defining the LOGR policy. See [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).

The return and reason codes for LOGR parameter errors are displayed in system message IXG002E or IXG447I. These messages are documented in the section [IXGINVNT – Managing the LOGR inventory couple data set](#) of the [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).

The requests you can specify for LOGR include:

- DEFINE LOGSTREAM
- DEFINE STRUCTURE
- UPDATE LOGSTREAM
- DELETE LOGSTREAM
- DELETE STRUCTURE
- LIST LOGSTREAM
- LIST STRUCTURE
- CONTINUE

The following restrictions apply when defining, updating, deleting, or listing a log stream or structure for the sysplex scope active system logger couple data set type LOGR:

- If the System Authorization Facility (SAF) is available, the system performs SAF authorization checks on all IXGINVNT requests and IXCMIAPU TYPE(LOGR) requests.

To define or delete a structure entry in the LOGR policy, the caller must have alter access to RESOURCE(MVSADMIN.LOGR) in SAF class CLASS(FACILITY).

To define, update, or delete a log stream entry in the LOGR policy, the caller must have alter access to RESOURCE(log_stream_name) in SAF class CLASS(LOGSTRM).

If the log stream entry specifies the STRUCTNAME keyword, then the caller must also have update access authority to the coupling facility structure through RESOURCE(IXLSTR.structure_name) in SAF class CLASS(FACILITY).

If you use the LIKE keyword to model a log stream's definition after another log stream, and you specified STRUCTNAME (structure_name) when you defined the model log stream, then you must also have update access to the RESOURCE (IXLSTR.structure_name) in class CLASS(FACILITY).

- If you use the ZAI(YES) keyword specification on a log stream's definition or update, then the caller must have update access to the RESOURCE (IXGZAWARE_CLIENT) in class CLASS(FACILITY).
- To list a log stream or structure entry from the LOGR policy, the caller must have read access to RESOURCE(MVSADMIN.LOGR) in SAF class CLASS(FACILITY).
- Be aware that if SAF is not available or if there is no CLASS(LOGSTRM) or CLASS(FACILITY) class defined for the log stream or structure, no security checking is performed. System Logger resources can be read and altered if the appropriate security definitions do not exist.

REPORT(YES|NO)

Indicates whether the utility should list out the LOGR policy. The default is REPORT(YES). IXCMIAPU generates the report after all control statements have been processed, regardless of whether errors were encountered during the processing of the control statements.

Note: To run a report, the caller must have read access to RESOURCE(MVSADMIN.LOGR) in SAF class CLASS(FACILITY).

DEFINE LOGSTREAM keywords and parameters

The DEFINE LOGSTREAM specification requests that an entry for a log stream is to be defined in the LOGR policy. The keywords are explained as follows:

[GROUP=(PRODUCTION|TEST)]

An optional keyword input that specifies whether the log stream is in the test group or the production group. This keyword allows you to keep processing and resources for log streams in the two groups separate on a single system, including requests such as data set allocation and data set recalls.

If the TEST group fails, the failure does not normally affect the PRODUCTION group. The active primary TYPE=LOGR couple data set in the sysplex must be formatted at a z/OS Version 1 Release 2 (HBB7705) or higher format level in order to specify the GROUP keyword. Otherwise, the request will fail with a return code 8, reason code 0839.

If you specify GROUP(PRODUCTION), which is the default, system logger places this log stream in the PRODUCTION group. A PRODUCTION log stream can use at least 75% of the system logger couple data set DSEXTENT records and connection slots.

If you specify GROUP(TEST), system logger places this log stream in the TEST group. TEST log streams are limited to at most 25% of the system logger couple data set DSEXTENT records and connection slots.

The GROUP value you specify must match the group setting for the structure that the log stream is being defined for, because system logger does not allow you to define a mixture of TEST and PRODUCTION log streams to a single structure. When you define the first log stream to a structure, the structure becomes either a TEST or PRODUCTION structure. After that, the GROUP value for subsequent log streams defined to a structure must match the GROUP value of the initial log stream. For example, if you specify or default to GROUP(PRODUCTION) for the first log stream defined to a structure, you will only be able to define PRODUCTION log streams to that structure subsequently.

NAME(streamname)

Specifies the name of the log stream that you want to define in the LOGR policy. The name can be made up of one or more segments separated by periods, up to the maximum length of 26 characters. The following rules apply:

- Each segment can contain up to eight numeric, alphabetic, or national (\$, #, or @) characters.
- The first character of each segment must be an alphabetic or national character.
- Each segment must be separated by periods, which you must count as characters.

RMNAME(NO_RMNAME)

RMNAME(rmname)

Specifies the name of the resource manager program associated with the log stream. If you specify RMNAME(NO_RMNAME), which is the default, the macro will be invoked as if RMNAME was not specified. RNAME must be 8 alphanumeric or national (\$, #, or @) characters, padded on the right with blanks if necessary.

You must define RMNAME in the LOGR policy before the resource manager can connect to the log stream.

See the system logger chapter in *z/OS MVS Programming: Assembler Services Guide* for information about writing a resource manager program to back up a log stream.

DESCRIPTION(description)

Specifies the user defined data describing the log stream. DESCRIPTION must be 16 alphanumeric or national (\$, #, @) characters, underscore (_) or period (.), padded on the right with blanks if necessary.

DASDONLY(NO)

DASDONLY(YES)

Specifies whether the log stream being defined is a coupling facility or a DASD-only log stream.

If you specify DASDONLY(NO), which is the default, the log stream is defined as a coupling facility log stream. With DASDONLY(NO), you can also specify STG_DUPLEX, DUPLEXMODE, and LOGGERDUPLEX keyword to select a method of duplexing for a coupling facility log stream.

If you specify DASDONLY(YES) the log stream is defined as a DASD-only log stream and does not use the coupling facility for log data.

Because a staging data set is required when using a DASD-only log stream, check the usage of the STG_SIZE parameter, the STG_DATACLAS parameter, or the defaults used for sizing the staging data set.

DASD-only log streams are unconditionally duplexed to staging data sets. This means that DASD-only log streams are created as if STG_DUPLEX(YES), DUPLEXMODE(UNCOND), and LOGGERDUPLEX(UNCOND) were specified when the log stream was defined. You cannot change these duplexing parameters. However, you can specify STG_DUPLEX(YES), DUPLEXMODE(UNCOND), and LOGGERDUPLEX(UNCOND). If you specify any other parameters for these keywords when you define a DASD-only log stream, the define request will fail. These keywords are optional.

With DASDONLY(NO), you can specify STG_DUPLEX, DUPLEXMODE, and LOGGERDUPLEX keywords to select a method of duplexing for a coupling facility log stream.

STRUCTNAME(structname)

Specifies the up to 16 character name of the coupling facility structure associated with the coupling facility log stream being defined. The structure specified is a list structure defined in the CFRM policy where all of this log stream's log blocks will be written before being written to DASD.

For a coupling facility log stream, you must define STRUCTNAME in the log stream definition in the LOGR policy via this parameter or the STRUCTNAME defined for the log stream referenced by the LIKE parameter before you can connect to the log stream.

For a DASD-only log stream, omit the STRUCTNAME parameter, since there is no coupling facility associated with the log stream.

STRUCTNAME must be 16 alphanumeric or national (\$, #, @) characters, or underscore (_), padded on the right with blanks if necessary. The first character must be alphabetic.

MAXBUFSIZE(maxbufsize)

MAXBUFSIZE(65532)

Specifies the size, in bytes, of the largest log block that can be written to the DASD-only log stream being defined in this request.

The value for MAXBUFSIZE must be between 1 and 65,532 bytes. The default is 65,532 bytes.

You can only specify the MAXBUFSIZE on a structure definition for a DASD-only log stream. For a coupling facility structure, specify MAXBUFSIZE in the structure definition.

STG_DUPLEX(NO)

STG_DUPLEX(YES)

Specifies whether the log stream data for a coupling facility log stream should be duplexed in DASD staging data sets.

For coupling facility log streams:

The default is STG_DUPLEX(NO). If you specify or default to STG_DUPLEX(NO), log data for a coupling facility log stream will be duplexed in local buffers, which might be vulnerable to system failure if your configuration contains a single point of failure.

If you specify STG_DUPLEX(YES), log data for a coupling facility log stream will be duplexed in staging data sets when the conditions defined by the DUPLEXMODE keyword are met. This method safeguards data on DASD staging data sets.

You can use the DUPLEXMODE keyword with STG_DUPLEX and with LOGGERDUPLEX to specify the type of duplexing desired and whether you want conditional or unconditional duplexing by System Logger.

For DASD-only log streams:

You can either omit this keyword or specify STG_DUPLEX(YES). In either case, log data will be unconditionally duplexed to staging data sets. Specifying any other parameter for the STG_DUPLEX keyword will result in an error for DASD-only log streams. See the LOGGERDUPLEX keyword for additional duplexing options.

DUPLEXMODE(COND)

DUPLEXMODE(UNCOND)

Specifies the conditions under which the log data for a log stream should be duplexed in DASD staging data sets.

For coupling facility log streams:

The default is DUPLEXMODE(COND). If you specify or default to DUPLEXMODE(COND), the coupling facility log data will be duplexed in staging data sets only if a system's connection to the coupling facility log stream contains a single point of failure and is therefore vulnerable to permanent log data loss:

- A connection to a log stream contains a single point of failure if the coupling facility is volatile or resides on the same CPC as the MVS system connecting to it. The coupling facility log data for the system connection containing the single point of failure will be duplexed to staging data sets.
- A connection to a log stream is failure-independent when the coupling facility for the log stream is non-volatile and resides on a different central processor complex (CPC) than the MVS system connecting to it. The coupling facility log data for that system connection will not be duplexed to staging data sets.

If you specify DUPLEXMODE(UNCOND), the log data for the coupling facility log stream will be duplexed in staging data sets, unconditionally, even if the connection is failure independent.

Note: If DUPLEXMODE (DRXRC) is specified, the log data is duplexed same as if the COND option is specified. This means that in staging data sets, if a system's connection to the coupling facility log stream contains a single point of failure, it is therefore vulnerable to permanent log data loss. For more information, see the COND DUPLEXMODE option.

You can use the DUPLEXMODE keyword with STG_DUPLEX and with LOGGERDUPLEX to specify the type of duplexing desired and whether you want conditional or unconditional duplexing by System Logger. See [“Selecting a method of duplexing coupling facility log data” on page 283](#) for complete information about using staging data sets to duplex coupling facility log data.

DUPLEXMODE is valid only when STG_DUPLEX(YES) is also specified.

Note: The staging data set related keywords, STG_SIZE, STG_DATACLAS, STG_MGMTCLAS, and STG_STORCLAS will remain set for the log stream and be used for any dynamic staging data set allocation during local recovery even after the conversion to STG_DUPLEX=NO.

For DASD-only log streams:

You can either omit this keyword or specify DUPLEXMODE(UNCOND). In either case, log data will be unconditionally duplexed to staging data sets. Specifying any parameter other than UNCOND for the DUPLEXMODE keyword will result in error for DASD-only log streams.

LOGGERDUPLEX(UNCOND)

LOGGERDUPLEX(COND)

Specifies whether System Logger will continue to provide its own log data duplexing, or conditionally not provide its own duplexing based on an alternative duplexing configuration that provides an equivalent or better recoverability of the log data.

For coupling facility log streams, the default parameter is LOGGERDUPLEX(UNCOND).

The active primary TYPE=LOGR couple data set in the sysplex must be formatted at a z/OS Release 2 or higher level in order to specify this keyword. Otherwise, the request will fail with a return code 8, reason code 0839.

For coupling facility log streams:

LOGGERDUPLEX(UNCOND) indicates that System Logger will provide its own specific duplexing of the log data regardless of any other duplexing (such as system-managed duplexing rebuild) that may be occurring.

LOGGERDUPLEX(COND) indicates that logger will provide its own specific duplexing of the log data unless the log stream is in an alternative duplexing configuration that provides an equivalent or better recoverability of the log data. For example, System Logger will not provide its own duplexing of the log data in the following configuration:

- When the log stream is in a non-volatile coupling facility list structure that is in a system-managed duplexing rebuild process (duplex mode)
- There is a failure independent relationship between the two structure instances, and
- There is a failure independent connection between connecting system and composite structure view.

See [“Logger and coupling facility duplexing combinations” on page 286](#) and [“System logger recovery” on page 300](#) for additional considerations on using the LOGGERDUPLEX keyword.

See Case 5 in [Table 23 on page 287](#) for additional details about the above configuration.

For DASD-only log streams:

You can either omit this keyword or specify the default of `LOGGERDUPLEX(UNCOND)`. In either case, log data will be unconditionally duplexed to staging data sets. Specifying any other parameter for the `LOGGERDUPLEX` keyword will result in an error for DASD-only log streams.

`STG_DATACLAS(NO STG_DATACLAS)`

`STG_DATACLAS(stg_dataclas)`

Specifies the up to 8-byte name of the SMS data class that will be used for allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you specify `STG_DATACLAS(NO_STG_DATACLAS)`, which is the default, the data class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the `STG_DATACLAS` parameter, including `NO_STG_DATACLAS`, **always** overrides one specified on a model log stream used on the `LIKE` parameter.

`STG_MGMTCLAS(NO STG_MGMTCLAS)`

`STG_MGMTCLAS(stg_mgmtclas)`

Specifies the up to 8-byte name of the SMS management class for the allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you specify `STG_MGMTCLAS(NO_STG_MGMTCLAS)`, which is the default, the management class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the `STG_MGMTCLAS` parameter, including `NO_STG_MGMTCLAS`, **always** overrides one specified on a model log stream used on the `LIKE` parameter.

`STG_STORCLAS(NO STG_STORCLAS)`

`STG_STORCLAS(stg_storclas)`

Specifies the up to 8-byte name of the SMS storage class for allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you specify `STG_STORCLAS(NO_STG_STORCLAS)`, which is the default, the storage class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the `STG_STORCLAS` parameter, including `NO_STG_STORCLAS`, **always** overrides one specified on a model log stream used on the `LIKE` parameter.

`STG_SIZE(stg_size)`

Specifies the size, in 4K blocks, of the DASD staging data set for the log stream being defined. The actual data set size of your data set depends on many factors including track size, CI SIZE, and volume type, and may be smaller or larger than your parameter inputs expect. See [“Testing log data set parameter modifications” on page 269](#) for important notes on choosing a data set size.

Specifying `STG_SIZE` overrides the space allocation attribute on the `STG_DATACLAS` parameter, if specified.

If you omit `STG_SIZE` for a coupling facility log stream, system logger does one of the following, in the order listed, to allocate space for staging data sets:

- Uses the `STG_SIZE` of the log stream specified on the `LIKE` parameter, if specified.
- Uses the maximum coupling facility structure size for the structure to which the log stream is defined. This value is obtained from the value defined on the `SIZE` parameter for the structure in the CFRM policy.

If you omit `STG_SIZE` for a DASD-only log stream, system logger does one of the following, in the order listed, to allocate space for staging data sets:

- Uses the `STG_SIZE` of the log stream specified on the `LIKE` parameter, if specified.
- Uses the size defined in the SMS data class for the staging data sets.
- Uses dynamic allocation rules for allocating data sets, if SMS is not available.

As of z/OS Version 2 Release 3, system logger supports staging data set sizes greater than 4GB. Use the STG_DATACLAS log stream specification and define the corresponding DFSMS data class with space allocation attributes and include in the data class definition the following attributes:

- Data Set Name Type "EXT" (extended format)
- Extended Addressability "Y"

Extended format is a means of storing data on a logical DASD volume. Extended addressability is a means to allow VSAM Data Sets greater addressability beyond the 4GB address constraint.

For more information about planning and sizing for staging data sets, see [“Plan space for staging data sets”](#) on page 263 and [“Set up the SMS environment for DASD data sets”](#) on page 266.

LS_ALLOCAHEAD(xls_allocahead | *)

Specifies whether offload data sets should be proactively allocated and opened in advance on systems connected to the log stream. It is an optional full-word input. The xls_allocahead value can be from 0 to 3.

The active primary TYPE=LOGR couple data set must be formatted at an HBB7705 or higher format level to specify this keyword. Otherwise, the request fails with return code 8 and reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more information.

When the value is 0 (default), logger does not proactively allocate and open advanced-current log stream offload data sets on any systems that are connected to the log stream.

When the value is between 1 and 3 (inclusively), all systems that are connected to and performing offloads for the log stream should be proactive in newly allocating up to the intended number (xls_allocahead) of advanced-current offload data sets. It also indicates these systems are proactive in opening the current offload data set as well as the first advanced-current offload data set. The logger processing is potentially affected on a system with the ALLOCAHEAD(NO) IXGCNFxx parmlib policy specification. See the IXGCNFxx parmlib member in *z/OS MVS Initialization and Tuning Reference* for more information about the MANAGE OFFLOAD ALLOCAHEAD specification. Also see [“Offloading log data from interim storage by freeing and/or moving it to DASD”](#) on page 222 for more information about the logger behavior based on the ALLOCAHEAD and LS_ALLOCAHEAD parameters.

The default is * 0. Omitting the parameter LS_ALLOCAHEAD results in the log stream being defined with an advanced-current allocate ahead value of zero (0) unless the LIKE parameter is also specified. If the LIKE parameter is specified, the LS_ALLOCAHEAD value is copied from the referenced LIKE log stream entry.

LS_DATACLAS(NO_LS_DATACLAS)

LS_DATACLAS(ls_dataclas)

Specifies the up to 8-byte name of the SMS data class that will be used for log stream offload data set allocation. The first character must be an alphabetic or national character.

If you specify LS_DATACLAS(NO_LS_DATACLAS), which is the default, the data class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the LS_DATACLAS parameter, including NO_LS_DATACLAS, **always** overrides one specified on a model log stream used on the LIKE parameter.

LS_MGMTCLAS(NO_LS_MGMTCLAS)

LS_MGMTCLAS(ls_mgmtclas)

Specifies the up to 8-byte name of the SMS management class for the log stream offload data set allocation. The first character must be an alphabetic or national character.

If you specify LS_MGMTCLAS(NO_LS_MGMTCLAS), which is the default, the management class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the LS_MGMTCLAS parameter, including NO_LS_MGMTCLAS, **always** overrides one specified on a model log stream used on the LIKE parameter.

LS_STORCLAS(NO_LS_STORCLAS)

LS_STORCLAS(ls_storclas)

Specifies the up to 8-byte name of the SMS storage class for the log stream offload data set allocation. The first character must be an alphabetic or national character.

If you specify LS_STORCLAS(NO_LS_STORCLAS), which is the default, the storage class is defined by standard SMS processing. See *z/OS DFSMS Using Data Sets* for more information about SMS. An SMS value specified on the LS_STORCLAS parameter, including NO_LS_STORCLAS, **always** overrides one specified on a model log stream used on the LIKE parameter.

LS_SIZE(ls_size)

Specifies the size, in 4K blocks, of the log stream offload DASD data sets for the log stream being defined.

The actual data set size of your data set depends on many factors including track size, CI SIZE, and volume type, and may be smaller or larger than your parameter inputs expect. See [“Testing log data set parameter modifications” on page 269](#) for important notes on choosing a data set size.

Specifying LS_SIZE overrides the space allocation attribute on the LS_DATACLAS parameter, if it was specified.

If you omit LS_SIZE, system logger does one of the following to allocate offload data sets:

- Uses the LS_SIZE of the log stream specified on the LIKE parameter, if specified.
- Uses the size defined in the SMS data class for the offload data sets.
- Uses dynamic allocation rules for allocating data sets, if SMS is not available.

AUTODELETE(NO)

AUTODELETE(YES)

Specifies when system logger can physically delete log data.

If you specify AUTODELETE(NO), which is the default, system logger physically deletes an entire log data set only when **both** of the following are true:

- Data is marked for deletion by a system logger application using the IXGDELET service or an archiving procedure (like CICS/VR for CICS log streams or IFBEREPS, which is shipped in SYS1.SAMPLIB, for logrec log streams).
- The retention period for all the data in the log data set has expired.

If you specify AUTODELETE(YES), system logger automatically physically deletes log data whenever data is either marked for deletion (using the IXGDELET service or an archiving procedure) or the retention period for all the log data in a data set has expired.

Be careful when using AUTODELETE(YES) if the system logger application manages log data deletion using the IXGDELET service. With AUTODELETE(YES), system logger may delete data that the application expects to be accessible. If you specify AUTODELETE=YES with RETPD=0, data is eligible for deletion as soon as it is written to the log stream.

The LOGR couple data set must be formatted at the OS/390 Release 3 level or above to use this keyword.

RETPD(0)

RETPD(retpd)

Specifies the retention period, in days, for log data in the log stream. The retention period begins when data is written to the log stream. Once the retention period for an entire log data set has expired, the data set is eligible for physical deletion. The point at which system logger physically deletes the data depends on what you have specified on the AUTODELETE parameter.

System logger processes RETPD when a log data set fills and system logger switches to a new one for a log stream. System logger will not process a retention period or delete data on behalf of log streams that are not connected and being written to by an application.

The value specified for RETPD must be between 0 and 65,536.

HLQ(NO HLQ)

HLQ(hlq)

Specifies the up to 8-byte high-level qualifier for both the log stream data set name and the staging data set name. HLQ must be 8 alphanumeric or national (\$,#,or @) characters, padded on the right with blanks if necessary. The first character must be an alphabetic or national character.

If you do not specify a high-level qualifier, or if you specify HLQ(NO_HLQ), the log stream will have a high-level qualifier of IXGLOGR. If you specified the LIKE parameter, the log stream will have the high-level qualifier of the log stream specified on the LIKE parameter. The value specified for HLQ overrides the high-level qualifier for the log stream specified on the LIKE parameter.

HLQ and EHLQ are mutually exclusive and cannot be specified for the same log stream definition.

If the name specified for the HLQ parameter refers to a field that contains X'00', then the macro will be invoked as if NO_HLQ had been specified. However, specifying HLQ=NO_HLQ and EHLQ=*ehlq* on the same request results in an error. When HLQ=NO_HLQ is specified, then the resulting high-level qualifier will be determined by the EHLQ value from the LIKE log stream or using a default value.

EHLQ(NO_EHLQ)

EHLQ(*ehlq*)

Specifies the name (or address in a register) of a 33-byte input field containing the extended high-level qualifier for both the log stream data set name and the staging data set name.

Syntax requirements for the extended high-level qualifier are as follows:

- The extended high-level qualifier must be 33 alphanumeric or national (\$, #, or @) characters, padded on the right with blanks if necessary.
- The value can be made up of one or more qualifiers (each 1 to 8 characters) separated by periods, up to the maximum length of 33 characters.
- Each qualifier must contain up to eight alphabetic, national, or numeric characters. Lowercase alphabetic characters will be folded to uppercase.
- The first character of each qualifier must be an alphabetic or national character.
- Each qualifier must be separated by a period, which you must count as a character.
- The resulting length of concatenating the significant characters from the EHLQ value with the STREAMNAME value (including the period delimiter) cannot exceed 35 characters.

EHLQ and HLQ are mutually exclusive and cannot be specified for the same log stream definition.

When the EHLQ parameter is not explicitly specified on the request, the resulting high-level qualifier to be used for the log stream data sets will be based on whether the HLQ or LIKE parameters are specified. If the HLQ parameter is specified, then that value will be used for the log stream data sets. When no high-level qualifier is explicitly specified on the DEFINE LOGSTREAM request, but the LIKE parameter is specified, then the high-level qualifier value being used in the referenced log stream will be used for the newly defined log stream. If the EHLQ, HLQ, and LIKE parameters are not specified, then the default value "IXGLOGR" will be used.

When EHLQ=NO_EHLQ is specified, then the resulting high-level qualifier will be determined by the HLQ value from the LIKE log stream or using a default value.

The active primary TYPE=LOGR couple data set must be formatted at a z/OS release 1.2 or higher level in order to specify the EHLQ keyword. Otherwise, the request will fail with return code 8, reason code X'0839'.

Examples:

1. Assume the OPERLOG log stream (NAME=SYSPLEX.OPERLOG) is defined with "EHLQ=MY.OWN.PREFIX" specified. The log stream data set would be allocated as follows, where the suffix is up to an eight-character field provided by System Logger.

```
MY.OWN.PREFIX.SYSPLEX.OPERLOG.suffix
```

2. Assume the OPERLOG log stream (NAME=SYSPLEX.OPERLOG) is attempted to be defined with "EHLQ=MY.PREFIX.IS.TOO.LONG". Even though the EHLQ value is less than the maximum 33 characters, a log stream data set cannot be allocated with an overall name greater than 44 characters. In this example, the define request would fail with return code 8, reason code IXLRNSCODEBADEHLQ. This data set name is not valid because it is too long.

```
MY.PREFIX.IS.TOO.LONG.SYSPLEX.OPERLOG.suffix
```

HIGHOFFLOAD(80)**HIGHOFFLOAD(highoffload)**

Specifies the percent value you want to use as the high offload threshold for the coupling facility space allocated for this log stream. When the coupling facility is filled to the high offload threshold point or beyond, system logger begins offloading data from the coupling facility to the DASD log stream data sets.

The default HIGHOFFLOAD value is 80%. You can specify the default in the following ways:

- HIGHOFFLOAD(80)
- HIGHOFFLOAD(0)
- Omit the HIGHOFFLOAD parameter.

The value specified for HIGHOFFLOAD must be greater than the LOWOFFLOAD value.

LOWOFFLOAD(0)**LOWOFFLOAD(lowoffload)**

Specifies the percent value you want to use as the low offload threshold for the coupling facility for this log stream. The low offload threshold is the point in the coupling facility, in percent value of space consumed, where system logger stops offloading coupling facility log data to log stream DASD data sets. The value of LOWOFFLOAD is the percent of log data system logger leaves in the coupling facility.

If you specify LOWOFFLOAD(0), which is the default, or omit the LOWOFFLOAD parameter, system logger uses the 0% usage mark as the low offload threshold.

The value specified for LOWOFFLOAD must be less than the HIGHOFFLOAD value.

WARNPRIMARY(NO_WARNPRIMARY)**WARNPRIMARY(NO)****WARNPRIMARY(YES)**

Optional keyword input that specifies whether monitoring warning messages should be issued based on the log stream primary (interim) storage consumption above the HIGHOFFLOAD value.

The active primary TYPE=LOGR couple data set must be formatted at an HBB7705 (or higher) format level in order to specify WARNPRIMARY=YES. Otherwise, the request will fail with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

If you omit the WARNPRIMARY parameter, the result will be the same as if you coded the NO_WARNPRIMARY option.

NO_WARNPRIMARY

Indicates that the WARNPRIMARY(NO) attribute should be used for the log stream unless the LIKE parameter is specified. If the LIKE parameter is specified, the WARNPRIMARY value will be copied from the referenced LIKE log stream entry.

NO

Indicates that the primary storage consumption monitoring warning messages will not be issued for the log stream.

YES

Indicates that log stream monitoring warning messages should be issued for the following conditions:

- When the log stream primary (interim) storage consumption is 2/3 between the HIGHOFFLOAD value and 100% full (rounded down to the nearest whole number). This is called the log stream imminent alert threshold.

For example, assume the default value is used for the HIGHOFFLOAD percentage. That would mean a HIGHOFFLOAD value of 80 would be used, so the warning message would be triggered for this case at $(2(100 - 80)/3 + 80) = 93\%$.

- For a CF based log stream, when a 90% entry full condition is encountered.
- When an interim (primary) storage full condition is encountered.

For more details on the messages and monitoring primary storage consumption, see [“Monitoring log stream interim storage consumption”](#) on page 228.

Note: This value can be overridden on the system level by system logger parameter options for IXGCNF keyword CONSUMPTIONMALERT(SUPPRESS).

LIKE(NO LIKE)**LIKE(like_streamname)**

Specifies the name of a log stream defined in the LOGR policy. The characteristics of this log stream (such as storage class, management class, high level qualifier and data class) will be copied for the log stream you are defining only if those characteristics are not explicitly coded on the referencing log stream. The parameters explicitly coded on this request, however, override the characteristics of the log stream specified on the LIKE parameter.

MODEL(NO)**MODEL(YES)**

Specifies whether the log stream being defined is a model, exclusively for use with the LIKE parameter to set up general characteristics for other log stream definitions.

If you specify MODEL(NO), which is the default, the log stream being defined is not a model log stream. Systems can connect to and use this log stream. The log stream can also be specified on the LIKE parameter, but is not exclusively for use as a model.

If you specify MODEL(YES), the log stream being defined is only a model log stream. It can be specified only as a model for other log stream definitions on the LIKE parameter.

Programs **cannot** connect to a log stream name that is defined as a model (MODEL(YES)) using an IXGCONN request.

No log stream data sets are allocated on behalf of a model log stream.

The attributes of a model log stream are syntax checked at the time of the request, but not verified until a another log stream references the model log stream on the LIKE parameter.

DIAG(NO)**DIAG(YES)**

Specifies whether or not dumping or additional diagnostics should be provided by System Logger for certain conditions.

If you specify DIAG(NO), which is the default, this indicates no special System Logger diagnostic activity is requested for this logstream regardless of the DIAG specifications on the IXGCONN, IXGDELET and IXGBRWSE requests.

If you specify DIAG(YES), this indicates that special Logger diagnostic activity is allowed for this log stream:

- Informational LOGREC software symptom records (denoted by RETCODE VALU/H00000004) as well as other external alerts highlighting suboptimal conditions. See the section on enabling additional log stream diagnostics in [z/OS MVS Diagnosis: Reference](#) for more details.
- When the appropriate specifications are provided on IXGCONN, IXGDELET, or IXGBRWSE requests by the exploiting application, further additional diagnostics may be captured. See the section on dumping on data loss (804-type) conditions in [z/OS MVS Programming: Assembler Services Guide](#) for more details.

OFFLOADRECALL(YES)**OFFLOADRECALL(NO)**

Indicates whether offload processing is to recall the current offload data set.

If you specify OFFLOADRECALL(YES), offload processing should recall the current offload data set.

If you specify OFFLOADRECALL(NO), offload processing should skip recalling the current offload data set and allocate a new one. Note that this option may cause any or all of the current offload data set to be wasted space on DASD once it is recalled. Care should be taken when using this option to size the data sets appropriately.

With OFFLOADRECALL(NO), system logger will request that allocation not wait on any ENQ serialization contention to be resolved, and will receive a class two type error (unavailable system resource), as described in [Interpreting DYNALLOC return codes in z/OS MVS Programming: Authorized Assembler Services Guide](#).

ZAI(NO)**ZAI(YES)****ZAI(NO_ZAI)**

Optional keyword that specifies whether the actual log stream data should be sent to the IBM zAware server.

The active primary TYPE=LOGR couple data set must be formatted at a z/OS V1R2 or later release level in order to specify ZAI(YES). Otherwise, the request will fail with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

If you omit the ZAI parameter, the result will be the same as if you coded the NO_ZAI option.

If you specify ZAI(NO), it indicates that the log stream data will not be included in data sent from this z/OS IBM zAware log stream client to the IBM zAware server. NO is the default.

If you specify ZAI(YES), it indicates that the log stream data will be included in data sent to the IBM zAware server providing the z/OS IBM zAware log stream client is established. See [“Preparing for z/OS IBM zAware log stream client usage”](#) on page 311 for more details on establishing and using z/OS IBM zAware log stream clients. Also refer to ZAIDATA keyword.

If you specify ZAI(NO_ZAI), it indicates that the default ZAI(NO) attribute should be used for the log stream unless the LIKE parameter is specified. If the LIKE parameter is specified, the ZAI value will be copied from the referenced LIKE log stream entry. Exercise care to ensure any newly defined log streams do not have the ZAI(YES) designation unless that is the absolute intention.

ZAIDATA('NO_ZAIDATA')**ZAIDATA('Xzaidata')**

Optional keyword that provides the value, if any, to be passed to the IBM zAware server when the z/OS IBM zAware log stream client is established (refer to ZAI(YES) keyword specification). The ZAIDATA value must be between 1 and 48 characters long and enclosed in apostrophes.

The value you specify for the ZAIDATA keyword is defined by and must match the intended log data type and capabilities of the IBM zAware server. See [“Preparing for z/OS IBM zAware log stream client usage”](#) on page 311 for more details on establishing and using z/OS IBM zAware log stream clients.

The active primary TYPE=LOGR couple data set must be formatted at z/OS V1R1 or later level in order to specify the ZAIDATA keyword option. Otherwise, the request will fail with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

If you specify ZAIDATA('NO_ZAIDATA'), it indicates that a null value will be used for log stream ZAIDATA attribute. NO_ZAIDATA is the default.

If you specify ZAIDATA(Xzaidata), it indicates the value that will be passed to the IBM zAware server when the z/OS IBM zAware log stream client is established.

The Xzaidata value must be between 1 and 48 characters long and enclosed in apostrophes.

The enclosing apostrophes are not counted as part of the value length. For example, assume the specification ZAIDATA('OPERLOG'). Even though 9 characters are specified within the keyword parenthesis, only 7 characters are within the delimiting apostrophes. So the resulting Xzaidata value length would be 7 for the string OPERLOG.

To code an apostrophe as part of the Xzaidata value, code 2 apostrophes where needed; it will result in 1 apostrophe and only count as 1 character in the value length. For example, assume the following specification of 26 characters within the keyword parenthesis ZAIDATA('OPERLOG,"somedata",END'). Because there are 24 characters within the delimiting apostrophes, and 2 apostrophes are coded twice within the string, the resulting Xzaidata value has 22 significant characters: OPERLOG,'somedata',END.

Valid characters are alphanumeric or national (\$, #, @) characters, and any of the special (graphical) characters listed in Table 32 on page 370. Lower case alphabetic characters will be folded to uppercase. Any other character will be converted into an EBCDIC blank (X'40').

<i>Table 32. Valid special (graphical) characters</i>		
Character Name	Symbol	Hexidecimal (EBCDIC)
EBCDIC blank	<blank>	X'40'
Cent sign	¢	X'4A'
Period	.	X'4B'
Less than sign	<	X'4C'
Left parenthesis	(X'4D'
Plus sign	+	X'4E'
Or sign		X'4F'
Ampersand	&	X'50'
Exclamation point	!	X'5A'
Asterisk	*	X'5C'
Right parenthesis)	X'5D'
Semicolon	;	X'5E'
Not sign	¬	X'5F'
Minus sign (hyphen)	-	X'60'
Slash	/	X'61'
Comma	,	X'6B'
Percent Sign	%	X'6C'
Underscore	_	X'6D'
Greater than sign	>	X'6E'
Question mark	?	X'6F'
Emphasis mark	`	X'79'
Colon	:	X'7A'
Apostrophe	'	X'7D'
Equal sign	=	X'7E'
Quote	"	X'7F'
Tilde	~	X'A1'
Left brace	{	X'C0'
Right brace	}	X'D0'
Backslash	\	X'E0'

If the resulting Xzaidata parameter value contains all X'40' (blanks), the ZAIDATA keyword will be treated as if NO_ZAIDATA has been specified.

Default: NO_ZAIDATA

If you omit the ZAIDATA parameter, the default will be used unless the LIKE parameter is specified. If the LIKE parameter is specified, the ZAIDATA value will be copied from the referenced LIKE log stream entry.

If you specify the ZAIDATA parameter, the value always overrides one specified on a model log stream used on the LIKE parameter.

DEFINE STRUCTURE keywords and parameters

The DEFINE STRUCTURE specification requests that an entry for a coupling facility structure be defined in the LOGR policy for a coupling facility log stream. The keywords are explained as follows:

NAME(structname)

Specifies the up to 16-byte name of the coupling facility structure you are defining. NAME must be 16 alphanumeric or national (\$, #, @) characters, or underscore (_), padded on the right with blanks if necessary. The first character must be alphabetic.

LOGSNUM(logsnum)

Specifies the number of log streams that can be allocated in the coupling facility list structure that is being defined. Note that this is the number of logstreams in the structure, not the number of connections to the structure. *logsnum* must be a value between 0 and 512.

LOGSNUM is required when defining a structure.

MAXBUFSIZE(maxbufsize)

Specifies the size, in bytes, of the largest log block that can be written to log streams allocated to the coupling facility specified in this request.

The value for MAXBUFSIZE must be between 1 and 65532 bytes. The default is 65532 bytes.

AVGBUFSIZE(avgbuFSIZE)

Specifies the average size in bytes, of log blocks written to all the log streams using this coupling facility structure.

System logger uses the average buffer size to control the entry-to-element ratio for this coupling facility structure.

When the active primary LOGR couple data set is at an OS/390 Release 3 level or higher, system logger uses the AVGBUFSIZE specified simply to make an initial determination of the entry-to-element ratio for the structure. After that, system logger monitors structure usage and dynamically manages the entry-to-element ratio accordingly. System logger uses the last entry-to-element ratio in effect for a structure for subsequent structure reallocation requests.

When the active primary LOGR couple data set is at a pre-OS/390 Release 3 level, system logger uses the AVGBUFSIZE specified to calculate an entry-to-element ratio that lasts for the life of this coupling facility structure. You cannot update the average buffer size for a structure without first deleting the structure definition (and all the log stream definitions associated with the structure) and then redefining the structure with a new average buffer size.

The *avgbuFSIZE* must be between 1 and the value for MAXBUFSIZE. The default value is 1/2 of the MAXBUFSIZE value.

UPDATE LOGSTREAM keywords and parameters

The UPDATE LOGSTREAM specification requests that an entry for a coupling facility or DASD-only log stream be updated in the LOGR policy. See [“Updating a log stream's attributes” on page 296](#) for information about which attributes are eligible for updating. The keywords are explained as follows:

NAME(name)

Specifies the name of the log stream that you want to update in the LOGR policy. NAME is required when updating a log stream.

NEWSTREAMNAME({newstreamname|NO_NEWSTREAMNAME})

Specifies a new name for the log stream identified in the NAME parameter. With this keyword, you can maintain the current data in a log stream under a new name and get new work going after timely defining a new instance of the log stream with the original name.

The new log stream name must be 26 characters long, padded on the right with blanks if necessary. Lowercase alphabetic characters are folded to uppercase. The name is made up of one or more

segments, up to the maximum length of 26 characters. Each segment may validly contain 1-8 numeric characters, alphabetic characters, national (\$, #, @) characters. Segments are joined by periods. The first character of each segment must be an alphabetic or national (\$, #, @) character.

For detailed description about this keyword, see [“Renaming a log stream dynamically”](#) on page 299.

Omitting the NEWSTREAMNAME parameter does not affect the current name of the log stream. If NO_NEWSTREAMNAME is specified for NEWSTREAMNAME, the macro is invoked as if the NEWSTREAMNAME parameter was not specified.

DEFAULT: NO_NEWSTREAMNAME

GROUP=(PRODUCTION|TEST)

An optional keyword input that lets you specify whether the log stream is in the test group or the production group. This keyword allows you to keep processing and resources for log streams in the two groups separate on a single system, including requests such as data set allocation and data set recalls. If the TEST group fails, the failure does not normally affect the PRODUCTION group. You can only specify the GROUP parameter on an UPDATE request when:

- The LOGR couple data set for the sysplex is formatted at the z/OS V1R2 level or higher.
- The structure has no log streams, failed or active, connected. (The request fails with return code 8, reason code X'0810' if there are connectors to the structure.)

If you specify GROUP(PRODUCTION), System Logger places this log stream in the PRODUCTION group. A PRODUCTION log stream can use at least 75% of the system logger couple data set DSEXTENT records and connection slots.

If you specify GROUP(TEST), system logger places this log stream in the TEST group. TEST log streams are limited to at most 25% of the system logger couple data set DS EXTENT records and connection slots.

The GROUP value that you specify must match the group setting for the structure that the log stream is being defined for, because system logger does not allow you to define a mixture of TEST and PRODUCTION log streams to a single structure. When you define the first log stream to a structure, the structure requires the GROUP value of this first log stream. After that, the GROUP value for subsequent log streams defined to a structure must match the GROUP value of the initial log stream. For example, if you specify or default to GROUP(PRODUCTION) for the first log stream defined to a structure, you are only able to define PRODUCTION log streams to that structure subsequently. If the GROUP value that is associated with the structure does not match with the subsequently defined log streams, the message IXG002E with the reason code 08, return code 8E9 (IxgRsnCodeBadGroup), are displayed.

If you update a log stream from PRODUCTION to TEST, this might affect the DS EXTENT records allocation since the limit changes from 75% to 25% of the system logger couple data set DS EXTENT records and connection slots. The system issues message IXC270I indicating that there is a shortage of DS EXTENT records for TEST log streams. If you update a log stream from TEST to PRODUCTION, more DS EXTENTS are available to the log stream and message IXG270I might be DOMed.

To update the log stream GROUP type for an existing structure, you need to take one of the following actions:

- Remove all of the log streams from the structure and define a log stream of the desired type to the structure. Only log streams of the same group type can be assigned to the structure.
- Issue an update request against the last remaining log stream in the structure for a group to change it to the desired GROUP. If there is only one log stream in a structure, you can update the GROUP value for the log stream and that of the structure.

For examples of changing the GROUP value, see [“Upgrading an existing log stream configuration”](#) on page 293 in *z/OS MVS Setting Up a Sysplex*.

STRUCTNAME(structname)

With UPDATE LOGSTREAM specifies the name of the coupling facility structure where all of this log stream's log blocks are written before being offloaded to DASD. This keyword can only be specified

to upgrade a DASD-only log stream to a coupling facility log stream. The structure specified is a list structure defined in the CFRM policy.

When the active primary LOGR couple data set in the sysplex is formatted at a z/OS Version 1 Release 2 (HBB7705) or higher format level, this keyword can be specified for a log stream that is currently structure-based to upgrade the log stream to a different coupling facility structure. If the LOGR couple data set is not formatted at the appropriate level, the request fails with a return code 8, reason code X'0839'.

STRUCTNAME must be 16 alphanumeric or national (\$, #, or @) characters, or underscore (_), padded on the right with blanks if necessary. The first character must be alphabetic.

Note that the MAXBUFSIZE value in the structure definition for this structure must be equal to or greater than the MAXBUFSIZE specified in the DASD-only log stream definition you are upgrading to a coupling facility log stream. For example, if the MAXBUFSIZE value in the DASD-only log stream definition you are upgrading is 50,000 bytes. When you specify a structure name to upgrade the DASD-only log stream to a coupling facility log stream, make sure that you specify one with a MAXBUFSIZE value of 50,000 bytes or more.

RMNAME(rmname)

Specifies the name of the resource manager program associated with the log stream. RNAME must be 8 alphanumeric or national (\$, #, or @) characters, padded on the right with blanks if necessary.

If you specify RMNAME(NO_RMNAME), system logger deletes the RMNAME defined for this log stream.

You must define RMNAME in the LOGR policy before the resource manager can connect to the log stream.

See the system logger chapter in *z/OS MVS Programming: Assembler Services Guide* for information about writing a resource manager program to back up a log stream.

DESCRIPTION(description)

Specifies the user defined data describing the log stream. DESCRIPTION must be 16 alphanumeric or national (\$, #, @) characters, underscore (_) or period (.), padded on the right with blanks if necessary.

MAXBUFSIZE(maxbufsize)

MAXBUFSIZE(65532)

Specifies the size, in bytes, of the largest log block that can be written to the DASD-only log stream being updated in this request. Use this parameter to update the MAXBUFSIZE for a DASD-only log stream.

The value for MAXBUFSIZE must be between 1 and 65,532 bytes and cannot be less than the current MAXBUFSIZE specified for the DASD-only log stream. (You can increase the MAXBUFSIZE, but you cannot decrease it.)

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. The change is immediately reflected in the log stream definition, but does not take affect until the subsequent first connection to the DASD-only log stream in the sysplex.

There is no default for the MAXBUFSIZE parameter on an UPDATE request. If you omit this parameter, there is no change to the MAXBUFSIZE for this log stream definition.

STG_DUPLEX(NO)

STG_DUPLEX(YES)

Specifies whether the log stream data for a coupling facility log stream may be duplexed in DASD staging data sets.

If you specify STG_DUPLEX(NO), log data for a coupling facility log stream is not be duplexed in DASD staging data sets, regardless of the failure independence/dependence of the coupling facility. The coupling facility resident log data will be duplexed in the local buffers of the z/OS image that wrote the data.

A coupling facility is considered failure independent when it is non-volatile and resides on a different CPC from the z/OS image using it. Otherwise, the coupling facility is failure dependent.

If you specify STG_DUPLEX(YES), the log data for a coupling facility log stream is duplexed when the conditions defined by the DUPLEXMODE keyword are fulfilled.

This keyword can be specified even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, the request fails with a return code 8, reason code X'0810'.

The change is immediately reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex or following a successful coupling facility user-managed structure rebuild.

For DASD-only log streams, you are allowed to specify STG_DUPLEX(YES), however, it has no effect on the log stream as DASD-only log streams are unconditionally duplexed to staging data sets. If you specify STG_DUPLEX(NO), the request fails unless you are upgrading a DASD-only log stream to a coupling facility log stream.

There is no default for the STG_DUPLEX keyword on an UPDATE request. If you omit this keyword, there will be no change to the staging duplexing status for this log stream definition.

You can use the DUPLEXMODE keyword with STG_DUPLEX and with LOGGERDUPLEX to specify the type of duplexing desired and whether you want conditional or unconditional duplexing by System Logger.

DUPLEXMODE(COND)

DUPLEXMODE(UNCOND)

Specifies the conditions under which log data for a coupling facility log stream should be duplexed in DASD staging data sets.

If you specify DUPLEXMODE(COND), the coupling facility log data is duplexed in staging data sets only if a system's connection to the coupling facility log stream contains a single point of failure and is therefore vulnerable to permanent log data loss:

- A connection to a log stream contains a single point of failure if the coupling facility is volatile and/or resides on the same central processing complex as the MVS system connection to it. The coupling facility log data for the system connection containing the single point of failure is duplexed to staging data sets.
- A connection to a log stream is failure-independent when the coupling facility for the log stream is non-volatile and resides on a different central processing complex than the MVS system connecting to it. The coupling facility log data for that system connection is not be duplexed to staging data sets.

If you specify DUPLEXMODE(UNCOND), the log data for the coupling facility log stream is duplexed in staging data sets, unconditionally, even if the coupling facility is failure-independent.

Note: If DUPLEXMODE (DRXRC) is specified, the log data is duplexed same as if the COND option is specified. This means that in staging data sets, if a system's connection to the coupling facility log stream contains a single point of failure, it is therefore vulnerable to permanent log data loss. For more information, see the COND DUPLEXMODE option.

Note: The staging data set keywords STG_SIZE, STG_DATACLAS, STG_MGMTCLAS, and STG_STORCLAS will remain set for the log stream and be used for any dynamic staging data set allocation during local recovery even after the conversion to STG_DUPLEX=NO.

This keyword can be specified even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 level. If a lower format level LOGR couple data set is being used, then the request fails with a return code 8, reason code X'0810'.

The change is immediately reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex or following a successful coupling facility user-managed structure rebuild.

There is no default for the DUPLEXMODE keyword on an UPDATE request. If you omit this keyword, there is no change to the duplexing mode for this log stream definition.

See *z/OS MVS Programming: Assembler Services Guide* for complete information about using staging data sets to duplex coupling facility log data.

You can use the DUPLEXMODE keyword with STG_DUPLEX and with LOGGERDUPLEX to specify the type of duplexing desired and whether you want conditional or unconditional duplexing by System Logger. See [“Selecting a method of duplexing coupling facility log data” on page 283](#) for incomplete information about using staging data sets to duplex coupling facility log data.

DUPLEXMODE is valid only when STG_DUPLEX(YES) has been specified for a coupling facility log stream.

For DASD-only log streams you are allowed to specify DUPLEXMODE(UNCOND), however, it has no effect on the log stream as DASD-only log streams are unconditionally duplexed to staging data sets. If you specify DUPLEXMODE(COND), the request fails unless you are upgrading a DASD-only log stream to a coupling facility log stream.

LOGGERDUPLEX(UNCOND)

LOGGERDUPLEX(COND)

Specifies whether System Logger continues to provide its own log data duplexing, or conditionally not provide its own duplexing based on an alternative duplexing configuration that provides an equivalent or better recoverability of the log data.

The active primary TYPE=LOGR couple data set in the sysplex must be formatted at a z/OS Release 2 or higher level to specify this keyword. Otherwise, the request fails with a return code 8, reason code 0839.

For DASD-only log streams, you are allowed to specify LOGGERDUPLEX(UNCOND), however, it has no effect on the log stream, as DASD-only log streams are already unconditionally duplexed to staging data sets. If you specify LOGGERDUPLEX(COND), the request fails unless you are upgrading a DASD-only log stream to a coupling facility log stream.

This keyword can be specified even when the log stream is actively connected when the LOGR couple data set is formatted at z/OS Release 2 or higher. If a lower format level LOGR couple data set is being used, then the request fails. The change is immediately reflected in the log stream definition. It will take effect on the subsequent first connection to the log stream in the sysplex or following a successful coupling facility user-managed structure rebuild.

Omitting the Update Logstream LOGGERDUPLEX keyword does not change how System Logger handles the duplexing of the log stream's log data.

See [“Logger and coupling facility duplexing combinations” on page 286](#) and [“System logger recovery” on page 300](#) for additional considerations on using the LOGGERDUPLEX keyword.

LOGGERDUPLEX(UNCOND) indicates that System Logger provides its own specific duplexing of the log data regardless of any other duplexing (such as system-managed duplexing rebuild) that may occur.

LOGGERDUPLEX(COND) indicates that logger provides its own specific duplexing of the log data unless the log stream is in an alternative duplexing configuration that provides an equivalent or better recoverability of the log data. For example, System Logger does not provide its own duplexing of the log data in the following configuration:

- When the log stream is in a non-volatile coupling facility list structure that is in a system-managed duplexing rebuild process (duplex mode)
- There is a failure independent relationship between the two structure instances, and
- There is a failure independent connection between connecting system and composite structure view.

See Case 5 in Table 23 on [page 287](#) for additional details about the LOGGERDUPLEX parameter and a coupling facility log stream.

HIGHOFFLOAD(highoffload)

Specifies the percent value you want to use as the high offload threshold for the coupling facility space allocated for this log stream. When the coupling facility is filled to the high offload threshold point or beyond, system logger begins offloading data from the coupling facility to the DASD log stream data sets.

IBM recommends that you are careful in considering to define your HIGHOFFLOAD value to greater than 80%. Defining a higher high offload threshold can leave you vulnerable to filling your coupling facility space for the log stream, which means that system logger rejects all write requests until the coupling facility log data can be offloaded to DASD log data sets.

The value specified for HIGHOFFLOAD must be higher than the LOWOFFLOAD value.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change is immediately reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. For a DASD-only log stream, the change also will take effect upon the next offload data set switch.

LOWOFFLOAD(lowoffload)

Specifies the percent value you want to use as the low offload threshold for the coupling facility for this log stream. The low offload threshold is the point in the coupling facility, in percent value of space consumed, where system logger stops offloading coupling facility log data to log stream DASD data sets. The value of LOWOFFLOAD is the percent of log data system logger leaves in the coupling facility.

There is no default for the LOWOFFLOAD parameter on an UPDATE request. If you omit this parameter, there is no change to the low offload value for this log stream definition.

The value specified for LOWOFFLOAD must be less than the HIGHOFFLOAD value.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change is immediately reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change also takes effect during the next structure rebuild. For a DASD-only log stream, the change will also take effect upon the next offload data set switch.

WARNPRIMARY(NO_WARNPRIMARY)**WARNPRIMARY(NO)****WARNPRIMARY(YES)**

Optional keyword input that specifies whether monitoring warning messages should be issued based on the log stream primary (interim) storage consumption above the HIGHOFFLOAD value.

The active primary TYPE=LOGR couple data set must be formatted at an HBB7705 (or higher) format level to specify WARNPRIMARY=YES. Otherwise, the request fails with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

This keyword can be updated even when the log stream is actively connected. The change is immediately reflected as a pending update in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change also takes effect during the next (user-managed or system-managed) structure rebuild. For a DASD-only log stream, the change will also take effect upon the next offload data set switch event.

NO_WARNPRIMARY

If you omit this keyword or specify the NO-WARNPRIMARY (default update) option, there is no change to the WARNPRIMARY specification for this log stream definition.

NO

Indicates that the primary storage consumption monitoring warning messages is not issued for the log stream.

YES

Indicates that log stream monitoring warning messages should be issued for the following conditions:

- When the log stream primary (interim) storage consumption is 2/3 between the HIGHOFFLOAD value and 100% full (rounded down to the nearest whole number). This is called the log stream imminent alert threshold.

For example, assume the default value is used for the HIGHOFFLOAD percentage. That would mean a HIGHOFFLOAD value of 80 would be used, so the warning message would be triggered for this case at $(2(100 - 80)/3 + 80) = 93\%$.

- For a CF based log stream, when a 90% entry full condition is encountered.
- When an interim (primary) storage full condition is encountered.

For more details on the messages and monitoring primary storage consumption, see [“Monitoring log stream interim storage consumption”](#) on page 228.

Note: This value can be overridden on the system level by system logger parameter options for IXGCNF keyword CONSUMPTIONMALERT(SUPPRESS).

STG_DATACLAS(stg_dataclas)

Specifies the name of the SMS data class that is used for allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you omit the STG_DATACLAS parameter, system logger makes no changes to the data class defined for this log stream.

If you specify STG_DATACLAS(NO_STG_DATACLAS) system logger deletes the STG_DATACLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. The changed STG_DATACLAS attribute will be used when a new staging data set needs to be created for the log stream.

STG_MGMTCLAS(stg_mgmtclas)

Specifies the name of the SMS management class for the allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you omit the STG_MGMTCLAS parameter, system logger makes no changes to the management class defined for this log stream.

If you specify STG_MGMTCLAS(NO_STG_MGMTCLAS) system logger deletes the STG_MGMTCLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. The changed STG_MGMTCLAS attribute will be used when a new staging data set needs to be created for the log stream.

STG_STORCLAS(stg_storclas)

Specifies the name of the SMS storage class for allocation of the DASD staging data set for this log stream. The first character must be an alphabetic or national character.

If you omit the STG_STORCLAS parameter, system logger makes no changes to the storage class defined for this log stream.

If you specify STG_STORCLAS(NO_STG_STORCLAS) system logger deletes the STG_STORCLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. The changed STG_STORCLAS attribute will be used when a new staging data set needs to be created for the log stream.

STG_SIZE(stg_size)

Specifies the size, in 4K blocks, of the DASD staging data set for the log stream being defined. The actual data set size of your data set depends on many factors including track size, CI SIZE, and volume type, and may be smaller or larger than your parameter inputs expect. See [“Testing log data set parameter modifications”](#) on page 269 for important notes on choosing a data set size.

Specifying STG_SIZE=0, will cause logger to use the SMS data class space allocation attribute of the new (if STG_DATACLAS is specified on the update) or existing STG_DATACLAS, or dynamic allocation rules, if STG_DATACLAS is not specified for dasdonly log streams, or the maximum structure size if STG_DATACLAS is not CF log streams.

Specifying STG_SIZE overrides the space allocation attribute on the STG_DATACLAS parameter, if STG_DATACLAS is specified on this update, a previous update, or define.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the HBB7705 format level. The change is immediately reflected in the log stream definition as a pending update. If a lower format level LOGR couple data set is being used, the request fails with return code 8, reason code X'0810'.

Pending updates take effect on the subsequent first connection to the log stream in the sysplex.

The changed STG_SIZE attribute will be used when a new staging data set needs to be created for the log stream.

For a structure-based log stream, the change is also reflected in the log stream definition during the next structure rebuild. Further action is necessary to cause the staging data set instance to be newly allocated with the new attributes for this case (for example, causing a last disconnect from the log stream on the z/OS system and subsequent first connection).

If you omit this parameter, there is no change to the DASD staging data size in this log stream definition. Note that if both the STG_DATACLAS and STG_SIZE are specified, the value for STG_SIZE overrides the space allocation attributes for the data class specified on the STG_DATACLAS value.

Omitting this parameter, causes the previous specification to remain in effect.

As of z/OS Version 2 Release 3, system logger supports staging data set sizes greater than 4GB. Use the STG_DATACLAS log stream specification and define the corresponding DFSMS data class with space allocation attributes and include in the data class definition the following attributes:

- Data Set Name Type "EXT" (extended format)
- Extended Addressability "Y"

Extended format is a means of storing data on a logical DASD volume. Extended addressability is a means to allow VSAM Data Sets greater addressability beyond the 4GB address constraint.

For more information about planning and sizing for staging data sets, see [“Plan space for staging data sets”](#) on page 263 and [“Set up the SMS environment for DASD data sets”](#) on page 266.

LS_ALLOCAHEAD(xls_allocahead | *)

Specifies whether offload data sets should be proactively allocated and opened in advance on systems connected to the log stream. It is an optional full-word input. The xls_allocahead value can be from 0 to 3.

The active primary TYPE=LOGR couple data set must be formatted at an HBB7705 or higher format level to specify this keyword. Otherwise, the request fails with return code 8 and reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more information.

When the value is 0, logger does not proactively allocate and open advanced-current log stream offload data sets on any systems that are connected to the log stream.

When the value is between 1 and 3 (inclusively), all systems that are connected to and performing offloads for the log stream should be proactive in newly allocating up to the intended number (xls_allocahead) of advanced-current offload data sets. It also indicates these systems are proactive in opening the current offload data set as well as the first advanced-current offload data set. The logger processing is potentially affected on a system with the ALLOCAHEAD(NO) IXGCNFxx parmlib policy specification. See the IXGCNFxx parmlib member in *z/OS MVS Initialization and Tuning Reference* for more information about the MANAGE OFFLOAD ALLOCAHEAD specification. Also see [“Offloading log data from interim storage by freeing and/or moving it to DASD”](#) on page 222 for more information about the logger behavior based on the ALLOCAHEAD and LS_ALLOCAHEAD parameters.

This keyword can be updated even when the log stream is actively connected. The change is immediately reflected as a pending update in the log stream definition. It takes effect when the next log stream offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change also takes effect during the next structure rebuild. The new value is used on the subsequent log stream offload activity after the value takes effect.

The default is *. Omitting the parameter LS_ALLOCAHEAD results in no change to the value for this log stream definition.

LS_DATACLAS(ls_dataclas)

Specifies the name of the SMS data class that is used for log stream offload data set allocation. The first character must be an alphabetic or national character.

If you omit the LS_DATACLAS parameter, system logger makes no changes to the data class defined for this log stream.

If you specify LS_DATACLAS(NO_LS_DATACLAS) system logger deletes the LS_DATACLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It will take effect when the next log stream offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild.

LS_MGMTCLAS(ls_mgmtclas)

Specifies the name of the SMS management class for the log stream offload data set allocation. The first character must be an alphabetic or national character.

If you omit the LS_MGMTCLAS parameter, system logger makes no changes to the management class defined for this log stream.

If you specify LS_MGMTCLAS(NO_LS_MGMTCLAS) system logger deletes the LS_MGMTCLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It will take effect when the next log stream offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild.

LS_STORCLAS(ls_storclas)

Specifies the name of the SMS storage class for the log stream offload data set allocation. The first character must be an alphabetic or national character.

If you omit the LS_STORCLAS parameter, system logger makes no changes to the storage class defined for this log stream.

If you specify LS_STORCLAS(NO_LS_STORCLAS) system logger deletes the LS_STORCLAS defined for this log stream.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the z/OS Release 2 format level. If a lower format level LOGR couple data set is being used, then the request fails. The change will immediately be reflected in the log stream definition. It will take effect when the next log stream offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild.

LS_SIZE(ls_size)

Specifies the size, in 4K blocks, of the log stream offload DASD data sets for the log stream being defined. The actual data set size of your data set depends on many factors including track size, CI SIZE, and volume type, and may be smaller or larger than your parameter inputs expect. See [“Testing log data set parameter modifications” on page 269](#) for important notes on choosing a data set size.

Specifying LS_SIZE=0 causes logger to use the SMS data class space allocation attribute of the new (if LS_DATACLASS is specified on the update) or existing LS_DATACLAS; or use dynamic allocation rules, if LS_DATACLAS is not specified.

Specifying LS_SIZE overrides the space allocation attribute on the LS_DATACLAS parameter, if LS_DATACLAS is specified on this update, the previous update, or define.

This keyword can be updated even when the log stream is actively connected when the LOGR couple data set is at least at the HBB7703 format level. If a lower format level LOGR couple data set is used, then the request fails with return code 8, reason code X'0810'. The change will immediately be reflected in the log stream definition. It will take effect when the next log stream offload data set is allocated (data set switch event) or on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild.

If this parameter is omitted, the previous specification to remain in effect.

AUTODELETE(NO)

AUTODELETE(YES)

Specifies when system logger can physically delete log data. This parameter can be specified regardless of whether the log stream is actively connected or not. The change will be immediately reflected in the log stream definition. It will take effect upon the next data set switch event or on the subsequent first connection to the log stream in the sysplex. To specify this keyword, the LOGR couple data set must be formatted at an OS/390 R3 level or higher; otherwise, the request fails.

If you specify AUTODELETE(NO), which is the default, system logger physically deletes an entire log data set only when **both** of the following are true:

- Data is marked for deletion by a system logger application using the IXGDELET service or an archiving procedure (like CICS/VR for CICS log streams or IFBEREPS, which is shipped in SYS1.SAMPLIB, for logrec log streams).
- The retention period for all the data in the log data set expires.

If you specify AUTODELETE(YES), system logger automatically physically deletes log data whenever data is either marked for deletion (using the IXGDELET service or an archiving procedure) or the retention period for all the log data in a data set has expired.

Be careful when using AUTODELETE(YES) if the system logger application manages log data deletion using the IXGDELET service. With AUTODELETE(YES), system logger may delete data that the application expects to be accessible. If you specify AUTODELETE=YES with RETPD=0, data is eligible for deletion as soon as it is written to the log stream.

The LOGR couple data set must be formatted at the OS/390 Release 3 level or above to use this keyword.

RETPD(0)**RETPD(retpd)**

Specifies the retention period, in days, for log data in the log stream. The retention period begins when data is written to the log stream. Once the retention period for an entire log data set has expired, the data set is eligible for physical deletion. The point at which system logger physically deletes the data depends on what you have specified on the AUTODELETE parameter. This parameter can be specified regardless of whether the log stream is actively connected or not. The change will be immediately reflected in the log stream definition. It will take effect upon the next data set switch event or on the subsequent first connection to the log stream in the sysplex. To specify this keyword, the LOGR couple data set must be formatted at an OS/390 R3 level or higher; otherwise, the request fails.

System logger processes RETPD when a log data set fills and system logger switches to a new one for a log stream. System logger will not process a retention period or delete data on behalf of log streams that are not connected and being written to by an application.

The value specified for RETPD must be between 0 and 65,536.

The active primary LOGR couple data set must be formatted at the OS/390 Release 3 level or above to use this keyword.

DIAG(NO DIAG)**DIAG(NO)****DIAG(YES)**

Specifies whether dumping or additional diagnostics should be provided by System Logger for certain conditions..

If you specify DIAG(NO_DIAG), which is the default, this indicates that the DIAG attribute of the log stream should not be updated.

If you specify DIAG(NO), this indicates no special System Logger diagnostic activity is requested for this logstream regardless of the DIAG specifications on the IXGCONN, IXGDELET and IXGBRWSE requests.

If you specify DIAG(YES), this indicates that special Logger diagnostic activity is allowed for this log stream:

- Informational LOGREC software symptom records (denoted by RETCODE VALU/H00000004) including other external alerts highlighting suboptimal conditions. See the section on enabling additional log stream diagnostics in *z/OS MVS Diagnosis: Reference* for more details.
- When the appropriate specifications are provided on IXGCONN, IXGDELET, or IXGBRWSE requests by the exploiting application, further additional diagnostics may be captured. See the section on dumping on data loss (804-type) conditions in *z/OS MVS Programming: Assembler Services Guide* for more details.

This keyword can be updated regardless of whether the log stream is actively connected or not. The change will immediately update the log stream definition but will not take effect on a particular system until the next first connection to the log stream from that system.

Omitting the DIAG parameter will not change the log stream's special diagnostics setting.

OFFLOADRECALL(NO_OFFLOADRECALL)**OFFLOADRECALL(YES)****OFFLOADRECALL(NO)**

Specifies whether offload processing is to recall the current offload data set.

This keyword can be updated even when the log stream is actively connected. The change will be immediately reflected in the log stream definition. It takes effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. For a DASD-only log stream, the change also will take effect upon the next offload data set switch.

If you specify OFFLOADRECALL(NO_OFFLOADRECALL), offload processing should not update the OffloadRecall attribute of the log stream.

If you specify OFFLOADRECALL(YES), offload processing should recall the current offload data set.

If you specify OFFLOADRECALL(NO), offload processing should not recall the current offload data set and allocate a new one. Note that this option may cause any or all of the current offload data set to be wasted space on DASD once it is recalled. Care should be taken when using this option to size the data sets appropriately.

With OFFLOADRECALL(NO), system logger will request that allocation not wait on any ENQ serialization contention to be resolved, and receives a class two type error (unavailable system resource), as described in [Interpreting DYNALLOC return codes in z/OS MVS Programming: Authorized Assembler Services Guide](#).

ZAI(NO_ZAI)

ZAI(NO)

ZAI(YES)

Optional keyword that specifies whether the actual log stream data should be sent to the IBM zAware server.

The active primary TYPE=LOGR couple data set must be formatted at z/OS V1R2 (HBB7705) or higher to specify ZAI(YES). Otherwise, the request fails with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

This parameter can be updated while there is an outstanding connection to the log stream. For this case, the change will immediately be reflected in the log stream definition. The updated specification takes effect on the following logger events for this log stream:

1. On the subsequent first connection to the log stream on a system,
2. As a result of a SETLOGR FORCE,ZAICONN,LSN= command on the target system, or
3. As a result of a SETLOGR FORCE,ZAICONN,ALL command when there is a connection to this log stream currently using the ZAI(YES) setting on the target system.

Note: Since the updated value can be used on a system by system basis, the installation should ensure the proper actions are taken on all systems with connections to the log stream to make use of the current value.

If you specify NO_ZAI, then you omit this keyword or specify the NO_ZAI (default) option. There is no change to the ZAI specification for this log stream definition.

If you specify NO, it indicates that the log stream data will not be included in the data sent from the z/OS IBM zAware log stream client to the IBM zAware server.

If you specify YES, it indicates that the log stream data will be included in data sent to the IBM zAware server provided the z/OS IBM zAware log stream client is established. Refer to [“Preparing for z/OS IBM zAware log stream client usage”](#) on page 311 for more details on establishing and using z/OS IBM zAware log stream clients. Also refer to the ZAIDATA keyword.

ZAIDATA('NO_ZAIDATA')

ZAIDATA('Xzaidata')

Optional keyword that provides the value, if any, to be passed to the IBM zAware server when the z/OS IBM zAware log stream client is established (refer to ZAI(YES) keyword specification). The ZAIDATA value must be between 1 and 48 characters long and enclosed in apostrophes.

The value you specify for the ZAIDATA keyword is defined by and must match the intended log data type and capabilities of the IBM zAware server. Refer to [“Preparing for z/OS IBM zAware log stream client usage”](#) on page 311 for more details on establishing and using z/OS IBM zAware log stream clients.

The active primary TYPE=LOGR couple data set must be formatted at the z/OS V1R2 (HBB7705) or higher level in order to specify the ZAIDATA keyword option. Otherwise, the request fails with return code 8, reason code X'0839'. See [“LOGR parameters for format utility \(sysplex scope\)”](#) on page 320 for more details.

This parameter can be updated while there is an outstanding connection to the log stream. For this case, the change will immediately be reflected in the log stream definition. The updated specification takes effect on the following logger events for this log stream:

1. On the subsequent first connection to the log stream on a system,
2. As a result of a SETLOGR FORCE,ZAICONN,LSN= command on the target system, or
3. As a result of a SETLOGR FORCE,ZAICONN,ALL command when there is a connection to this log stream currently using the ZAI=YES setting on the target system.

Note: Since the updated value can be used on a system by system basis, the installation should ensure the proper actions are taken on all systems with connections to the log stream to make use of the current value.

If you specify NO_ZAIDATA, it indicates that a null value is used for the log stream ZAIDATA attribute.

If you specify Xzaidata, it indicates the value that is passed to the IBM zAware server when the z/OS IBM zAware log stream client is established.

The Xzaidata value must be between 1 and 48 characters long and enclosed in apostrophes. The enclosing apostrophes are not counted towards the resulting value length. For example, assume the specification ZAIDATA('OPERLOG'). Even though 9 characters are specified within the keyword parenthesis, only 7 characters are within the delimiting apostrophes. So the resulting Xzaidata value length would be 7 for the string OPERLOG.

To code an apostrophe as part of the zaidata value, code 2 apostrophes where needed and it results in 1 apostrophe and only count as 1 character in the resulting value length. For example, assume the following specification of 26 characters within the keyword parenthesis ZAIDATA('OPERLOG,"somedata",END'). Since there are 24 characters within the delimiting apostrophes and 2 apostrophes are coded twice within the string, the resulting Xzaidata value would have 22 significant characters: OPERLOG,'somedata',END.

Valid characters are alphanumeric or national (\$, #, @) characters, and any of the special (graphical) characters listed Table 33 on page 383. Lowercase alphabetic characters will be folded to uppercase. Any other character will be converted into an EBCDIC blank (X'40').

Table 33. Valid special (graphical) characters		
Character Name	Symbol	Hexidecimal (EBCDIC)
EBCDIC blank	<blank>	X'40'
Cent sign	¢	X'4A'
Period	.	X'4B'
Less than sign	<	X'4C'
Left parenthesis	(X'4D'
Plus sign	+	X'4E'
Or sign		X'4F'
Ampersand	&	X'50'
Exclamation point	!	X'5A'
Asterisk	*	X'5C'
Right parenthesis)	X'5D'
Semicolon	;	X'5E'
Not sign	¬	X'5F'
Minus sign (hyphen)	-	X'60'
Slash	/	X'61'

Table 33. Valid special (graphical) characters (continued)		
Character Name	Symbol	Hexidecimal (EBCDIC)
Comma	,	X'6B'
Percent Sign	%	X'6C'
Underscore	_	X'6D'
Greater than sign	>	X'6E'
Question mark	?	X'6F'
Emphasis mark	`	X'79'
Colon	:	X'7A'
Apostrophe	'	X'7D'
Equal sign	=	X'7E'
Quote	"	X'7F'
Tilde	~	X'A1'
Left brace	{	X'C0'
Right brace	}	X'D0'
Backslash	\	X'E0'

If the resulting Xzaidata parameter value contains all X'40' (blanks), then the ZAIDATA keyword is treated as if NO_ZAIDATA is specified.

If you omit this keyword, there is no change to the ZAIDATA specification for this log stream definition.

DELETE LOGSTREAM keyword and parameters

The DELETE LOGSTREAM specification requests that an entry for a log stream be deleted from the LOGR policy. Note that you cannot delete a log stream while there are active connections to it (see [“Deleting log streams from the LOGR, LOGRY or LOGRZ policy”](#) on page 289). The keyword is explained as follows:

NAME(streamname)

Specifies the name of the log stream you want to delete from the LOGR policy. NAME is required when deleting a log stream.

DELETE STRUCTURE keyword and parameters

The DELETE STRUCTURE specification requests that an entry for a coupling facility structure be deleted from the LOGR policy. Note that you cannot delete a structure while there are active connections to any of the log streams defined to that structure. The keyword is explained as follows:

NAME(structname)

Specifies the name of the coupling facility structure you are deleting from the LOGR policy. NAME is required when deleting a structure.

See [“Deleting log streams from the LOGR, LOGRY or LOGRZ policy”](#) on page 289 for more details.

LIST LOGSTREAM keywords and parameters

The LIST LOGSTREAM specification requests that an entry for a log stream be listed in SYSPRINT output. The keywords are explained as follows:

NAME(streamname)

Specifies the name of the log stream you want to list. You will get the log stream definition from the LOGR policy in the output. NAME is required.

A log stream name can be specified with wildcards. See [“Using wildcards with the LIST request” on page 385](#).

DETAIL(NO)

DETAIL(YES)

Specifies whether you want detailed information about the log stream from the LOGR policy.

If you specify DETAIL(NO), which is the default, the output will only include the log stream definition.

If you specify DETAIL(YES) the output will include the log stream data as well as the definition.

LISTCAT

Specifies that LISTCAT output messages should be provided in the output for each log stream offload data set. Use this option when DETAIL(YES) is specified. If DETAIL(NO) is used, the LISTCAT parameter is ignored. When the LISTCAT parameter is used, system logger requests the equivalence of the following command from IDCAMS system services VSAM utility:

```
LISTCAT ENTRIES(log-stream-data-set-name) ALL
```

The combined logger and catalog information in the SYSPRINT output allows for easier determination of data sets that need attention or correction by the systems programmer. For samples and description of the output, see the [LISTCAT \(IDCAMS\) messages for offload data sets](#) topic in [z/OS MVS Diagnosis: Reference](#).

It is recommended that you use wildcards on the NAME keyword only if necessary because the LISTCAT option can produce a large amount of output for each log stream offload data set.

LIST STRUCTURE keywords and paramaters

The LIST STRUCTURE specification requests that an entry for a structure be listed in the SYSPRINT output. The keywords are explained as follows:

NAME(structname)

Specifies the name of the structure you want to list. You will get the structure definition from the LOGR policy in the output. NAME is required.

A structure name can be specified with wildcards. See [“Using wildcards with the LIST request” on page 385](#).

DETAIL(NO)

DETAIL(YES)

Specifies whether you want detailed information about the log stream from the LOGR policy.

If you specify DETAIL(NO), which is the default, the output will only include the structure definition.

If you specify DETAIL(YES), the output will include the structure data as well as the definition.

Using wildcards with the LIST request

You can use wildcards on LIST requests to select multiple log streams or structures in your SYSPRINT output. The wildcards you can use include:

Substitutes for zero or more characters, up to the maximum length of the string. An * can start the string, end it, appear in the middle or in multiple places in the string. A single * for the name will select all the structure or log stream names.

?

Substitutes for one character. One or more ? can start the string, end it, appear in the middle or in multiple places in the string.

Examples of using wildcards follow:

- *A* selects all names containing an A, including the name A.

- *A*B selects all names that contain an A followed by and ending with a B, with or without intervening characters. The names can have characters before the A.
- ?A? selects all 3-character names with an A as the second character.
- ?A?B selects all 4-character names with A as the second character and B as the fourth character.
- ?A* selects all names of 2 or more characters with A as the second character.

CONTINUE Keyword

The CONTINUE keyword allows you to continue processing TYPE=LOGR requests even if some earlier requests fail. This keyword applies only to the job step in which it appears as it does not span job steps. This keyword can be useful when you want to process many requests in a single job step and you want system logger to try to execute all the requests. Even if some fail.

The placement of the CONTINUE request is significant. If an error occurs in requests before the CONTINUE request is specified, system logger will not execute the remaining requests, including the CONTINUE request. However, system logger will perform syntax checking on the remaining statements as usual. If an error occurs after CONTINUE is specified, system logger will record the error and attempt to execute the remaining requests in the job step.

You only need to specify CONTINUE once. Subsequent CONTINUE requests are allowed, but will not produce a different effect than if CONTINUE was only specified once.

System logger displays message IXG004I for successful requests that follow the CONTINUE keyword. It displays messages IXG447I and IXG003I for failing requests that follow the CONTINUE keyword. The failure message IXG447I is similar to IXG002E, but does not mean system logger will stop trying to execute requests. At the end of the report, system logger displays message IXG446E if any request failed. This message presents the number of errors system logger found.

If your requests are contingent on a previous request, use a separate job steps and avoid the use of the CONTINUE request. The result of a job step with CONTINUE that has an error is an overall failure return code of 12.

In the event of a syntax error, the IXCMIAPU utility stops processing and will not execute the remaining requests, even if CONTINUE is specified.

For examples of the CONTINUE keyword, see [“Administrative data utility for the LOGR policy” on page 398](#). Also see the section titled [Utility error messages in z/OS MVS Diagnosis: Reference](#) for examples of message output.

LOGRY and LOGRZ keywords and parameters for the administrative data utility

System logger supports up to two single-system scope policies in a sysplex in addition to the sysplex scope LOGR policy. The LOGRY policy and the LOGRZ policy are each one of the two system logger policies that are single-system scope. You can define the policy information that describes the characteristics of a log stream that will be used when there are active connections to the log stream from a single system within the sysplex.

These log streams are separate resources from the log stream resources defined in the sysplex scope LOGR policy and the other single-system scope policy. You can specify the REPORT keyword for any of the requests.

When using a system logger sysplex scope LOGR policy in the sysplex, see [“LOGR keywords and parameters for the administrative data utility” on page 358](#). The LOGRY and LOGRZ policy information is similar to the LOGR policy information, except no structures can be defined and the log streams must be DASD-only type for the single-system policies.

The specific single-system scope LOGRY or LOGRZ couple data set must be activated and available before you add log stream definitions to it. See [“Using LOGRY or LOGRZ couple data sets for a single system scope within a sysplex” on page 279](#). When you intend to specify LOGRY or LOGRZ policy information, you must execute the IXCMIAPU utility on the system using the respective single-system scope policy.

For guidance on planning your system logger couple data set parameters, see [Chapter 10, “Planning for system logger applications,”](#) on page 213.

You can also use the IXGINVNT macro to define either the LOGRY or LOGRZ policy information. Note that the parameters of the administrative data utility are not identical to the IXGINVNT parameters for defining the LOGRY or LOGRZ policy. See [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).

The return and reason codes for LOGRY or LOGRZ policy parameter errors are displayed in system message IXG002E or IXG447I. These messages are documented in IXGINVNT – Managing the LOGR inventory couple data set of the [z/OS MVS Programming: Assembler Services Reference IAR-XCT](#).

The requests you can specify for LOGRY and LOGRZ policies include:

- DEFINE LOGSTREAM
- UPDATE LOGSTREAM
- DELETE LOGSTREAM
- LIST LOGSTREAM
- CONTINUE

The following restrictions apply when defining, updating, deleting, or listing a log stream for system logger:

- If the System Authorization Facility (SAF) is available, the system performs SAF authorization checks on all IXGINVNT requests and IXCMIAPU DATA TYPE(LOGRY) or TYPE(LOGRZ) requests.
To define, update, or delete a log stream entry in the LOGRY or LOGRZ policy, the caller must have alter access to RESOURCE(log_stream_name) in SAF class CLASS(LOGSTRM).
- If you use the ZAI(YES) keyword specification on a log stream's definition or update, then the caller must have update access to the RESOURCE(IXGZAWARE_CLIENT) in class CLASS(FACILITY).
- To list a log stream entry from the LOGRY or LOGRZ policy, the caller must have respective read access to RESOURCE(MVSADMIN.LOGRY) or RESOURCE(MVSADMIN.LOGRZ) in SAF class CLASS(FACILITY).
- Be aware that if SAF is not available or if there is no CLASS(LOGSTRM) or CLASS(FACILITY) class defined for the log stream, no security checking is performed. System Logger resources can be read and altered if the appropriate security definitions do not exist.

REPORT(YES|NO)

Indicates whether the utility should list out the LOGRY or LOGRZ policy. The default is REPORT(YES). IXCMIAPU generates the report after all control statements have been processed, regardless of whether errors were encountered during the processing of the control statements.

Note: To run a report, the caller must have respective read access to RESOURCE(MVSADMIN.LOGRY) or RESOURCE(MVSADMIN.LOGRZ) in SAF class CLASS(FACILITY).

DEFINE LOGSTREAM keywords and parameters

The DEFINE LOGSTREAM specification requests that an entry for a DASD-only log stream is to be defined in the active LOGRY or LOGRZ single-system scope policy for the system where the IXCMIAPU utility executes.

The specifications are the same as for the LOGR policy, except as noted in Table 34 on page 388. For the keyword details, see [“LOGR keywords and parameters for the administrative data utility”](#) on page 358.

Table 34. DEFINE LOGSTREAM keywords and parameters for single-system scope policy

DEFINE LOGSTREAM keyword	LOGR	LOGRY	LOGRZ	Comparison	Notes
NAME	yes	yes	yes	same	Log streams with the same name in different policies are distinct and separate resources within the sysplex.
GROUP	yes	yes	yes	same	
RMNAME	yes	yes	yes	same	
DESCRIPTION	yes	yes	yes	same	
DASDONLY	yes	yes	yes	same	Required for LOGRY and LOGRZ policies.
STRUCTNAME	yes	no	no	only allowed for LOGR policy	Not allowed for LOGRY and LOGRZ policies.
MAXBUFSIZE	yes	yes	yes	same	
STG_DUPLEX	yes	yes	yes	somewhat similar	For LOGRY and LOGRZ policies, STG_DUPLEX(NO) is not allowed since these policies must be for DASDONLY log stream.
DUPLEXMODE	yes	yes	yes	somewhat similar	For LOGRY and LOGRZ policies, DUPLEXMODE(UNCOND) is the only value allowed since these policies must be for DASDONLY log stream.
LOGGERDUPLEX	yes	yes	yes	somewhat similar	For LOGRY and LOGRZ policies, LOGGERDUPLEX(UNCOND) is the only value allowed since these policies must be for DASDONLY log stream.
STG_DATACLAS	yes	yes	yes	same	
STG_MGMTCLAS	yes	yes	yes	same	
STG_STORCLAS	yes	yes	yes	same	
STG_SIZE	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
LS_ALLOCAHEAD	yes	yes	yes	same	

Table 34. DEFINE LOGSTREAM keywords and parameters for single-system scope policy (continued)

DEFINE LOGSTREAM keyword	LOGR	LOGRY	LOGRZ	Comparison	Notes
LS_DATACLAS	yes	yes	yes	same	
LS_MGMTCLAS	yes	yes	yes	same	
LS_STORCLAS	yes	yes	yes	same	
LS_SIZE	yes	yes	yes	same	
AUTODELETE	yes	yes	yes	same	
RETPD	yes	yes	yes	same	
HLQ	yes	yes	yes	same, but with different defaults	For LOGR CDS, the default is IXGLOGR. For LOGRY CDS, the default is IXGLOGRY. For LOGRZ CDS, the default is IXGLOGRZ.
EHLQ	yes	yes	yes	same, but with different defaults	See HLQ.
HIGHOFFLOAD	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
LOWOFFLOAD	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
WARNPRIMARY	yes	yes	yes	same	
LIKE	yes	yes	yes	same	
MODEL	yes	yes	yes	same	
DIAG	yes	yes	yes	same	
ZAI	yes	yes	yes	same	
ZAIDATA	yes	yes	yes	same	

UPDATE LOGSTREAM keywords and parameters

The UPDATE LOGSTREAM specification requests that an entry for a DASD-only log stream is to be updated in the active LOGRY or LOGRZ single-system scope policy for the system where the IXCMIAPU utility executes.

See [“Updating a log stream's attributes” on page 296](#) for information about which attributes are eligible for updating.

The specifications are the same as for the LOGR policy, except as noted in Table 35 on page 390. For the keyword details, see [“LOGR keywords and parameters for the administrative data utility” on page 358](#).

Table 35. UPDATE LOGSTREAM keywords and parameters for single-system scope policy

UPDATE LOGSTREAM keyword	LOGR	LOGRY	LOGRZ	Comparison	Notes
NAME	yes	yes	yes	same	Identifies log stream to update in the active policy in use on the system where the utility executes.
NEWSTREAMNAME	yes	yes	yes	same	Log streams with the same name in different policies are distinct and separate resources within the sysplex.
GROUP	yes	yes	yes	same	
RMNAME	yes	yes	yes	same	
DESCRIPTION	yes	yes	yes	same	
STRUCTNAME	yes	no	no	can only change for LOGR policy	Not allowed for LOGRY and LOGRZ policies.
MAXBUFSIZE	yes	yes	yes	same	
STG_DUPLEX	yes	no	no	can only change for LOGR policy	For LOGRY and LOGRZ policies, STG_DUPLEX(NO) is not allowed since these policies must be for DASDONLY log stream.
DUPLEXMODE	yes	no	no	can only change for LOGR policy	For LOGRY and LOGRZ policies, DUPLEXMODE(UNCOND) is the only value allowed since these policies must be for DASDONLY log stream.
LOGGERDUPLEX	yes	no	no	can only change for LOGR policy	For LOGRY and LOGRZ policies, LOGGERDUPLEX(UNCOND) is the only value allowed since these policies must be for DASDONLY log stream.
STG_DATACLAS	yes	yes	yes	same	
STG_MGMTCLAS	yes	yes	yes	same	
STG_STORCLAS	yes	yes	yes	same	

Table 35. UPDATE LOGSTREAM keywords and parameters for single-system scope policy (continued)

UPDATE LOGSTREAM keyword	LOGR	LOGRY	LOGRZ	Comparison	Notes
STG_SIZE	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
LS_ALLOCAHEAD	yes	yes	yes	same	
LS_DATACLAS	yes	yes	yes	same	
LS_MGMTCLAS	yes	yes	yes	same	
LS_STORCLAS	yes	yes	yes	same	
LS_SIZE	yes	yes	yes	same	
AUTODELETE	yes	yes	yes	same	
RETPD	yes	yes	yes	same	
HIGHOFFLOAD	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
LOWOFFLOAD	yes	yes	yes	mostly similar	No coupling facility type log stream considerations.
WARNPRIMARY	yes	yes	yes	same	
LIKE	yes	yes	yes	same	
MODEL	yes	yes	yes	same	
DIAG	yes	yes	yes	same	
ZAI	yes	yes	yes	same	
ZAIDATA	yes	yes	yes	same	

DELETE LOGSTREAM keywords and parameters

The DELETE LOGSTREAM specification requests that an entry for a DASD-only log stream is to be deleted from the active LOGRY or LOGRZ single-system scope policy for the system where the IXCMIAPU utility executes.

Note that you cannot delete a log stream while there are active connections to it, see [“Deleting log streams from the LOGR, LOGRY or LOGRZ policy”](#) on page 289 for more details.

The specifications are the same as for the LOGR policy, except as noted in [Table 36 on page 391](#). See [“LOGR keywords and parameters for the administrative data utility”](#) on page 358 for the keyword details.

Table 36. DELETE LOGSTREAM keywords and parameters for single-system scope policy

DELETE LOGSTREAM keyword	LOGR	LOGRY	LOGRZ	Comparison	Notes
NAME	yes	yes	yes	same	Identifies log stream to delete from the active policy in use on the system where the utility executes.

SFM parameters for the administrative data utility

The SFM policy information refers to each system in the sysplex. In the policy, you can assign each system:

- A weight, to be used if signaling connectivity errors occur (WEIGHT parameter). The system also uses the assigned weights in conjunction with the REBUILDPERCENT value specified in the CFRM policy when determining whether to initiate a rebuild for a structure in a coupling facility to which a system has lost connectivity.
- An action to be taken when a status update missing condition occurs. You cannot specify SSUMLIMIT if you specify the PROMPT parameter.
- An interval after which SFM will remove a system that is status update missing but still producing XCF signalling traffic from the sysplex (SSUMLIMIT parameter). You can only specify SSUMLIMIT in conjunction with either DEACTTIME, RESETTIME, or ISOLATETIME.
- An interval after which SFM should take automatic action when signaling sympathy sickness occurs (MEMSTALLTIME).
- An interval after which the system should take automatic action to relieve a hang in a structure-related process (CFSTRHANGTIME).

The policy information also specifies whether SFM is to be used to automatically recover XCF signaling connectivity failures and what reconfiguration actions are to be taken when the PR/SM Automatic Reconfiguration Facility is being used (CONNFALL parameter).

CONNFALL(YES)

CONNFALL(NO)

Specifies whether or not the system is to perform recovery actions in the event of loss of signaling connectivity failures between one or more systems in the sysplex. Specifying YES indicates that the sysplex is to perform sysplex partitioning actions using the WEIGHT values specified for each system in the sysplex as a basis to determine the best set of systems to remain in the sysplex, given the connectivity failures that have occurred. Specifying NO indicates that the system is to prompt the operator to decide which system or systems to partition from the sysplex.

CONNFALL is an optional parameter. The default is YES.

SYSTEM

Specifies the definition of a system within the scope of the named SFM policy.

NAME(sysname)

NAME(*)

Specifies the system name. The limit for the number of systems that can be defined in a policy is established when the SFM couple data set is formatted. Use NAME(*) to specify policy default values for any system in the sysplex for which there is no explicit NAME statement or whose NAME statement does not explicitly specify one or more of the attributes on the NAME(*) statement. You can specify * only once per policy for the NAME parameter of the SYSTEM keyword. See [“Assigning defaults using an *” on page 396](#) for an example of using the * to specify installation default values in a policy.

sysname must be 1-8 characters. Valid characters are alphanumeric (A-Z and 0-9) and national characters (@, #, \$). The first character of *sysname* can be an alphanumeric or national character.

NAME is a required parameter.

WEIGHT(weight)

Specifies a value in the range from 1 to 9999 that represents the relative importance value of this particular system in the sysplex. This value is used in the determination of potential sysplex partitioning actions which would be required in the event of a signaling connectivity failure between one or more systems in the sysplex. This value is also used in conjunction with the REBUILDPERCENT value specified in the CFRM policy when determining whether MVS-initiated rebuild of coupling facility structures should take place when loss of connectivity to a coupling facility occurs.

WEIGHT is an optional parameter. The system default value is 1. You can change the default value for your policy by using the NAME(*) WEIGHT (your policy default value) option.

DEACTTIME(nostatus-interval)

Specifies the time interval in seconds after which the logical partition on which a failing system (identified by NAME) resides is to be deactivated. For example, specifying DEACTTIME(10) causes an LPAR DEACTIVATE of the failing LPAR 10 seconds after the system status update missing condition is detected for a system that is not producing XCF signalling traffic. This action is performed only if there is an active system in the sysplex running in LPAR mode on the same PR/SM CPC as the named system.

You can specify a time interval for DEACTTIME from 0 to 86400 seconds. Specifying a value of 0 implies that you want the system to deactivate the logical partition as soon as the status update missing condition is detected. You can set a default value for your installation by using the NAME(*) DEACTTIME(your default value) option.

Note:

1. If the failing system resumes its status update before the interval has expired, the deactivate function is not performed.
2. If the OPNOTIFY interval expires before the time specified for DEACTTIME, the IXC402D message will be issued before the deactivate is attempted.

RESETTIME(nostatus-interval)

Specifies the time interval in seconds after which the named system's PR/SM logical partition is to be system reset. For example, specifying RESETTIME(10) causes the failing system to be reset 10 seconds after a system status update missing condition is detected for a system that is not producing XCF signalling traffic. This action is performed only if there is an active system in the sysplex running in LPAR mode on the same PR/SM CPC as the named system.

You can specify a time interval for RESETTIME from 0 to 86400 seconds. Specifying RESETTIME(0) causes the failing system to be reset as soon as XCF detects the system status update missing condition. You can set a default value for your installation by using the NAME(*) RESETTIME(your default value) option.

Note:

1. If the failing system resumes its status update before the interval has expired, the system reset function is not performed.
2. If the OPNOTIFY interval expires before the time specified for RESETTIME, the IXC402D message is issued before the reset is attempted.

ISOLATETIME(nostatus-interval)

Specifies the time interval in seconds after which the named system is to be isolated using the fencing services through the coupling facility. For example, specifying ISOLATETIME(10) causes the failing system to be fenced 10 seconds after the System Status Update Missing condition is detected for a system that is not producing XCF signalling traffic. This action is performed only if there is an active system in the sysplex that shares connectivity to one or more coupling facilities with the named system.

You can specify a time interval for ISOLATETIME of from 0 to 86400 seconds. Specifying a value of 0 implies that you want the system to isolate the named system as soon as the status update missing condition is detected. You can set a default value for your installation by using the NAME(*) ISOLATETIME (your default value) option.

ISOLATETIME(0) is the default when none of the DEACTTIME, RESETTIME, PROMPT, or ISOLATETIME parameters is specified. IBM suggests using ISOLATETIME(0) to allow SFM to isolate and partition a failed system without operator intervention and without undue delay.

Note:

1. If the failing system resumes its status update before the interval has expired, the ISOLATE function is not performed.

2. If the isolation fails, message IXC102A prompts the operator to reset the system manually.

Coordinating Intervals: You should understand the relationship among several time intervals to which you assign a value.

- The INTERVAL specified in COUPLExx
- The spin loop timeout interval specified in EXSPATxx
- The RESETTIME, DEACTTIME, or ISOLATETIME interval described here

For information about the relationship between INTERVAL and SPINTIME values, see [“Planning the failure detection interval and operator notification interval” on page 178.](#)

SSUMLIMIT(NONE)

SSUMLIMIT(ssumlimit)

Specifies the time interval, in seconds, that a system can be in the status update missing condition but still producing XCF signals before SFM removes the system from the sysplex. For example, specifying SSUMLIMIT(10) prompts SFM to remove the failing system from the sysplex 10 seconds after SFM detects that the system is both in the status update missing condition but still producing XCF signals.

You cannot specify SSUMLIMIT if you specify the PROMPT parameter, because SSUMLIMIT processing is done in conjunction with one of the DEACTTIME, RESETTIME, or ISOLATETIME parameters. For example, if you specify the ISOLATETIME parameter but not the SSUMLIMIT parameter, the system default value of SSUMLIMIT(NONE) is assumed, regardless whether a policy default is specified. If you specify the SSUMLIMIT parameter but not one of the DEACTTIME, RESETTIME, or ISOLATETIME parameters, the system default value of ISOLATETIME(0) is assumed, regardless whether a policy default is specified.

You can specify a time for SSUMLIMIT from 0 to 86400 seconds or NONE, which is the default. The value specified for *ssumlimit* must be greater than or equal to the *nostatus-interval* specified on the DEACTTIME, RESETTIME, or ISOLATETIME:

- Specifying or defaulting to a SSUMLIMIT value of NONE implies that you do not want SFM to take automatic action against a system that is producing XCF signalling traffic.
- Specifying a SSUMLIMIT value equal to *nostatus_interval* implies that you want SFM to take automatic action against a system whether after *nosatus_interval*, regardless of it is producing XCF signalling traffic or not.
- Specifying a SSUMLIMIT value greater than *nostatus_interval* implies that you want SFM to take automatic action against a system that is producing XCF signalling traffic, but that SFM should wait longer than it would if the system was not producing XCF signalling traffic.

PROMPT

Specifies that an operator prompt will be issued if the named system enters a status update missing condition. The time interval used is the OPNOTIFY interval specified in COUPLExx.

MEMSTALLTIME(NO)

MEMSTALLTIME(seconds)

Indicates whether XCF is to take action to alleviate signaling sympathy sickness caused by a stalled XCF group member. When MEMSTALLTIME(NO) is specified, XCF will not take action to alleviate signaling sympathy sickness. Specify MEMSTALLTIME(seconds) to have XCF take action to alleviate signaling sympathy sickness. When signaling sympathy sickness is declared and persists for MEMSTALLTIME seconds, XCF will determine which stalled members are contributing to the problem. The member consuming the greatest amount of applicable signaling resource will be terminated. The MEMSTALLTIME value can be 0-64800 seconds. A value of at least 60 seconds is recommended to allow for dump processing.

MEMSTALLTIME is an optional parameter. The default is NO.

CFSTRHANGTIME(NO)

CFSTRHANGTIME(seconds)

Specifies the time interval, in seconds, that a coupling facility structure connector can remain unresponsive before the system takes action to relieve a hang in a structure-related process.

System actions may include stopping a rebuild, terminating the connector's task or address space, or partitioning the affected system. The system may initiate more than one of these actions, if the first action taken is not effective in relieving the hang.

The time interval is measured from the time that the system determines that a connector has failed to provide a required response, when IXL040E or IXL041E is issued (approximately 2 minutes after the event requiring the response was presented to the connector). For example, specifying CFSTRHANGTIME(300) causes the system to initiate action 5 minutes after IXL040E or IXL041E has been issued.

You can specify or default to CFSTRHANGTIME(NO), or specify a time between 0 and 1800 seconds.

- Specifying or defaulting to CFSTRHANGTIME(NO) means that the system will not take automatic action to relieve the hang. If the connector does not provide the required response, you must take manual action to allow the affected structure-related process to continue. It may be necessary to terminate the unresponsive connector.
- Specifying a value of 0 means that you want the system to initiate corrective action as soon as the connector is recognized as unresponsive.
- When you specify a numerical value, it controls both the time that may elapse before the system takes its first action, and the time that may elapse before it takes any subsequent escalation action that may be applicable. If the initial action is not effective, the system will wait the larger of CFSTRHANGTIME or 2 minutes before initiating the next applicable action.
- Specifying an out-of-range value will cause the policy definition to fail with message IXC730I.

RECONFIG

Specifies the LPAR reconfiguration action to be taken after a system has been removed from the sysplex through sysplex partitioning. The limit for the number of reconfiguration actions that can be defined for a system is established when the SFM couple data set is formatted.

FAILSYS(sysname)

Specifies the name of the failing system that has been partitioned out of the sysplex. Use of this parameter is intended to name the primary system responsible for processing a workload in the sysplex. Should this system fail, a PR/SM reconfiguration would allow another system in the sysplex to acquire the necessary resources to take over the workload processing responsibilities.

The *sysname* must be 1-8 characters. Valid characters are alphanumeric (A-Z and 0-9) and national characters (@,#\$). The first character of *sysname* can be an alphanumeric or national character.

FAILSYS is a required parameter.

ACTSYS(sysname)

Specifies the name of the "acting" system which is to perform the reconfiguration action in the sysplex. Use of this parameter is to name the system in the sysplex that would acquire the resources of the target system identified by TARGETSYS.

The logical partition identified by ACTSYS must be authorized to perform reconfiguration actions. See *PR/SM Planning Guide* for additional information about cross-partition control authority.

The *sysname* must be 1-8 characters. Valid characters are alphanumeric (A-Z and 0-9) and national characters (@,#\$). The first character of *sysname* can be an alphanumeric or national character.

ACTSYS is a required parameter.

Note: To reactivate TARGETSYS, storage resources that have been taken over by ACTSYS must be configured offline to ACTSYS before the TARGETSYS logical partition is reactivated.

TARGETSYS(sysname)

TARGETSYS(ALL)

Specifies whether the logical partition where the *sysname* resides is to be deactivated or that ALL other logical partitions in the same addressing range as the "acting" system are to be deactivated.

Note that deactivation of a specific system's logical partition can be performed only when both the target system and the acting system are running on the same PR/SM CPC and in the same sysplex.

Use of this parameter is intended to name the system (or all systems) that are to be deactivated so that the system identified by ACTSYS can take over its resources.

The *sysname* must be 1-8 characters. Valid characters are alphanumeric (A-Z and 0-9) and national characters (@,#\$). The first character of *sysname* can be an alphanumeric or national character.

TARGETSYS is a required parameter.

STORE(NO)

STORE(YES)

Specifies whether or not the acting system is to acquire central storage from potentially deactivated logical partitions.

If YES is specified and the LPAR deactivation is successful, the system issues the command to reconfigure the central storage. If YES is specified and the LPAR deactivation is not successful because the target LPAR is not active, the system still attempts to issue the storage reconfiguration command.

STORE is an optional parameter. The default is NO.

ESTORE(NO)

ESTORE(YES)

Specifies whether or not the acting system is to acquire expanded storage from potentially deactivated logical partitions.

If YES is specified, the system issues the command to reconfigure the expanded storage.

ESTORE is an optional parameter. The default is NO.

Assigning defaults using an *

Attributes specified on a SYSTEM NAME(*) statement apply to all systems for which there is no explicit SYSTEM NAME statement (that is, one with a name exactly matching that of the system), and to all systems whose NAME statements do not explicitly specify the attributes that are specified on the SYSTEM NAME(*) statement.

Note: For purposes of assigning attributes, the combination of indeterminate status action (ISOLATETIME, DEACTTIME, RESETTIME, or PROMPT) and SSUMLIMIT is considered to be a single attribute.

The example in [Figure 74 on page 396](#) shows the use of the * to assign installation default values in an SFM policy.

```
DEFINE POLICY NAME(POLICY1) ...
  SYSTEM NAME(SYS01)
    ISOLATETIME(100)
  SYSTEM NAME(SYS02)
    WEIGHT(25)
  SYSTEM NAME(*)
    WEIGHT(10)
    ISOLATETIME(0) SSUMLIMIT(150)
    CFSTRHANGTIME(300)
```

*Figure 74. Example: Using * to Assign Installation Defaults in SFM Policy*

In this example, system SYS01 uses the following parameter values:

- It requires the ISOLATETIME value of 100 seconds and accepts the system default of SSUMLIMIT(NONE). The SSUMLIMIT(150) specified on the SYSTEM NAME(*) statement does not apply to SYS01, because the SYSTEM NAME(SYS01) statement specifies an indeterminate status action of ISOLATETIME(100).

- It uses policy default WEIGHT value of 10 and CFSTRHANGTIME value of 300 established by the SYSTEM NAME(*) statement.
- It uses the system defaults for all attributes not specified on either the SYSTEM NAME(SYS01) or the SYSTEM NAME(*) statement, for example, MEMSTALLTIME(NO).

System SYS02 uses the following parameter values:

- It requires a WEIGHT value of 25.
- It uses the policy default combination of ISOLATETIME(0) SSUMLIMIT(150), and the policy default CFSTRHANGTIME value of 300 established by the SYSTEM NAME(*) statement.
- It uses the system defaults for all attributes not specified on either the SYSTEM NAME(SYS02) or the SYSTEM NAME(*) statement, for example, MEMSTALLTIME(NO).

All other systems use the following parameter values:

- The policy default combination of ISOLATETIME(0) SSUMLIMIT(150), the policy default WEIGHT value of 10, and the policy default CFSTRHANGTIME value of 300 established by the SYSTEM NAME(*) statement.
- The system defaults for all other attributes, for example, MEMSTALLTIME(NO).

Return codes

The return codes in decimal format from the administrative data utility are:

Value

Description

0

Successful completion. All requested processing completed successfully.

2

Requested processing completed successfully, but with warning conditions encountered.

4

An I/O error occurred on the SYSPRINT or SYSIN file, or the SYSIN file was empty. Some processing might have completed.

8

The SYSPRINT DD or the SYSIN DD was not specified or could not be used.

12

An error occurred parsing the input or the utility was unable to obtain the data space used for I/O processing.

16

The level of system is not at the correct level for running the utility.

20

An abend occurred during utility processing.

24

An environmental error occurred. The utility must not be linked to or called from another program.

Examples of using the IXCMIAPU utility

The z/OS examples shown in this topic are shipped with the product in SYS1.SAMPLIB.

Administrative data utility for automatic restart manager Policy Data

Sample JCL to run the utility to create or update the automatic restart management administrative data is shipped in SYS1.SAMPLIB member IXCARMPO.

Administrative data utility for CFRM policy data

Figure 75 on page 398 shows sample JCL to run the utility to create or update CFRM administrative data. The SYS1.SAMPLIB member name is IXCCFRMP.

```
//STEP20 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSIN DD *

DATA TYPE(CFRM) REPORT(YES)

DEFINE POLICY NAME(POLICY1) REPLACE(YES)

CF NAME(FACIL01)
  TYPE(123456)
  MFG(IBM)
  PLANT(PK)
  SEQUENCE(123456789012)
  PARTITION(01)
  DUMPSPACE(98M)

CF NAME(FACIL02)
  TYPE(123456)
  MFG(IBM)
  PLANT(PK)
  SEQUENCE(123456789012)
  PARTITION(2)
  DUMPSPACE(78M)

STRUCTURE NAME(LIST_01)
  SIZE(400M)
  INITSIZE(350M)
  MINSIZE(300M)
  SCMMAXSIZE(4G)
  SCMALGORITHM(KEYPRIORITY1)
  SCMMAXSIZE(4G)
  SCMALGORITHM(KEYPRIORITY1)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(85)
  PREFLIST(FACIL01,FACIL02)
  EXCLLIST(CACHE_01)

STRUCTURE NAME(CACHE_01)
  SIZE(1G)
  REBUILDPERCENT(25)
  DUPLEX(ENABLED)
  PREFLIST(FACIL02,FACIL01)
  EXCLLIST(LIST_01)

/* DELETE POLICY NAME(POLICY1) */
/*
```

Figure 75. IXCCFRMP - Administrative Policy Data in CFRM Couple Data Set

Administrative data utility for the LOGR policy

Sample JCL to run the utility to create or update the administrative policy in the sysplex scope primary and alternate couple data sets (LOGR policy) for the system logger service is shipped in SYS1.SAMPLIB member IXGLOGRP.

In the sample JCL:

```

STEP 1 defines a CF structure to logger, defines 3 log streams
(2 in the structure and 1 as DasdOnly), performs updates
on all 3 log streams, defines a 4th log stream as the
3rd one in the structure, deletes the 2nd log stream
definition, and then requests full listing reports for
log stream names matching wildcard name "STREAM*" and
for the specific structure name "STRUCTURE1".

STEP 2 requests logger continue deleting all the log streams
listed, even if one or more delete attempt is not
successful, and then request a full listing report for
any remaining log streams.
Note that the result of a job step with CONTINUE
that had an error but continued anyway, will have an
overall job step failure return code of 12.

STEP 3 defines a model log stream and defines a list of log
streams based on the model, only if the model log stream
define was successful, and then requests a full listing
report for all the log streams.

```

Figure 76. Example of running the utility to create or update the LOGR administrative policy

Administrative data utility for the LOGRY and LOGRZ policy data

Sample JCL to run the utility to create or update the administrative policy in a single-system scope primary and alternate couple data sets (either LOGRY or LOGRZ policy) for the system logger service is shipped in SYS1.SAMPLIB member IXGLOGZP.

In the sample JCL:

Steps 1 and 2 are samples for the DATA TYPE(LOGRZ) single-system scope policy.

They can be repeated using DATA TYPE(LOGRY) for the other single-system scope policy.

```

STEP 1 defines in the single-system scope active primary
and alternate LOGRZ couple data sets to logger 3 log
streams as DasdOnly, performs updates on all 3 log
streams, deletes the 2nd log stream definition, and
then requests full listing reports for log stream
names matching wildcard name "STREAM*".

STEP 2 defines a model log stream and defines a list of
log streams based on the model, only if the model
log stream define was successful, and then requests
a full listing report for all the log streams.

STEP 3 requests logger continue deleting all the log streams
listed, even if one or more delete attempt is not
successful, and then request a full listing report for
any remaining log streams.
Note that the result of a job step with CONTINUE that
had an error but continued anyway, will have an overall
job step failure return code of 12.

```

Figure 77. Example of running the utility to create or update the LOGR administrative policy

Administrative data utility for SFM policy data

Sample JCL to run the utility to create or update an SFM administrative policy is shown in [Figure 78 on page 400](#). The SYS1.SAMPLIB member name is IXCSFMP.

```
//STEP20 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSIN DD *

DATA TYPE(SFM) REPORT(YES)

DEFINE POLICY NAME(POLICY1) CONNFALL(NO) REPLACE(YES)

SYSTEM NAME(*)
ISOLATETIME(0) SSUMLIMIT(NONE)
MEMSTALLTIME(NO)
CFSTRHANGTIME(300)

DEFINE POLICY NAME(POLICY2) CONNFALL(YES) REPLACE(YES)

SYSTEM NAME(*)
ISOLATETIME(0) SSUMLIMIT(25)
WEIGHT(5)

SYSTEM NAME(SYS1)
PROMPT
WEIGHT(25)
MEMSTALLTIME(25)
CFSTRHANGTIME(NO)

DEFINE POLICY NAME(POLICY3) REPLACE(YES)

SYSTEM NAME(SYS1)
DEACTTIME(10)
WEIGHT(3)

SYSTEM NAME(SYS2)
DEACTTIME(10)
WEIGHT(10)

RECONFIG
FAILSYS(SYS1) ACTSYS(SYS2) TARGETSYS(ALL)
ESTORE(YES) STORE(YES)

/* DELETE POLICY NAME(POLICY1) */

/*
```

Figure 78. IXCSFMP - Administrative Policy Data in SFM Couple Data Set

Sample JCL to run the utility to create or update an SFM administrative policy is shipped in SYS1.SAMPLIB member IXCSFMP.

Chapter 13. Deletion utility for XCF group members

Use the IXCDELET utility to delete XCF group members that are created, quiesced, or failed. Sample JCL to run the utility is described here and is shipped in SYS1.SAMPLIB.

Use discretion when considering whether to use the IXCDELET utility. Do not use the utility unless you understand the use of the member by its owning XCF application.

A user of the IXCDELET utility must be defined with UPDATE access to the FACILITY CLASS resource MVSADMIN.XCF.IXCM2DEL.

Identify the XCF group member to be deleted by its group name and its member name. The member to be deleted must be in one of the following states: created, quiesced, or failed. If the member is not in one of those states, the utility program fails.

XCF group members can be deleted one at a time. To delete multiple members, the IXCDELET utility must be run multiple times.

Sample JCL to run the IXCDELET utility to delete created, quiesced, or failed XCF group members is shipped in SYS1.SAMPLIB member IXCDELUT.

The utility parameters are specified using a PARM= keyword on the EXEC PGM=IXCM2DEL statement that invokes the utility, as follows:

```
PARM='groupname,membername'
```

where groupname specifies the XCF group name of the member to be deleted and membername specifies the member name of the member to be deleted.

Output messages from the IXCDELET utility

The IXCDELET utility issues one or more of the following messages.

NOT AUTHORIZED FOR UPDATE BY MVSADMIN.XCF.IXCM2DEL RESOURCE

SAF authorization checking for update access to MVSADMIN.XCF.IXCM2DEL failed.

UNEXPECTED AUTHORIZATION RETURN CODE

SAF authorization checking for update access to MVSADMIN.XCF.IXCM2DEL received an unexpected return code.

XCF GROUP NAME LENGTH ERROR

The specified XCF group name exceeded the maximum length of a valid XCF group name.

XCF MEMBER NAME LENGTH ERROR

The specified XCF member name exceeded the maximum length of a valid XCF member name.

XCF GROUP/MEMBER NAME IMPROPERLY SPECIFIED

The XCF group and member name pair was improperly specified. For example, no comma delimiter was found to separate a group name from a member name.

INPUT PARAMETERS NOT SPECIFIED, OR TOO LONG

No PARM keyword was coded to pass the XCF group and member name as input, or the length of the PARM that was passed in was too large to consist of an XCF group and member name.

PROCESSING XCF GROUP grpname MEMBER memname

The IXCM2DEL utility successfully acquired the XCF group and member name indicated in the message from the input PARM keyword specification. The utility will attempt to query and delete the indicated target group/member. Subsequent message(s) indicate the result of this query and delete processing.

XCF MEMBER DELETION SUCCESSFUL

The target group/member was deleted.

XCF MEMBER NOT FOUND

The target group/member was not found by query.

XCF MEMBER NOT DEFINED

The target group/member was found by query, but was not defined at the time deletion was attempted. It must have become undefined during the window between the time of query and time of attempted deletion.

XCF MEMBER CANNOT BE DELETED

The target group/member is not in the correct state to be deleted. Only created, quiesced, or failed members can be deleted.

RESERVED FIELD ERROR

The delete request failed due to a nonzero reserved field in the parameter list.

INVALID ALET

The delete request failed due to an invalid parameter list ALET specification.

INVALID PARAMETER LIST VERSION NUMBER

The delete request failed due to an invalid parameter list version number.

INVALID FUNCTION CODE

The delete request failed due to an invalid parameter list function code.

INACCESSIBLE PARAMETER LIST

The delete request failed due to an inaccessible parameter list.

NOT IN TASK MODE

The delete request failed because the caller was not in task mode.

NOT ENABLED

The delete request failed because the caller was not enabled.

UNKNOWN PARAMETER ERROR

The delete request failed due to an unknown parameter error.

ISSUING TASK ABNORMAL TERMINATION

The delete request failed because the requesting task terminated abnormally during processing.

UNKNOWN ENVIRONMENTAL ERROR

The delete request failed due to an unknown environmental error.

XCF COMPONENT ERROR

The delete request failed due to an XCF component error.

XCF MEMBER NAME MISMATCH

The target group/member was found by query, but its group/member name no longer matched the input specification.

Chapter 14. Deletion utility for XCF note pads

Use the IXCDELNP utility to delete an XCF note pad that is defined in a Parallel Sysplex.



CAUTION: Take care when considering whether to use the IXCDELNP utility. Do not use the utility unless you understand the use of the note pad by its exploiting application and any adverse effects that the application might experience as a result.

A user of the IXCDELNP utility must have the proper SAF authority to run the IXCDELNP utility and the proper SAF authority to delete the target note pad.

UPDATE access authority to the MVSADMIN.XCF.IXCMINPD FACILITY class resource is required to run the IXCDELNP utility. CONTROL access authority to the note pad resource is required for this utility to successfully process deletion of a note pad. See [“Authorizing XCF note pad requests” on page 147](#) for information about defining security profiles for note pad requests.

The utility parameters are specified with a PARM= keyword on the EXEC PGM=IXCMINPD statement that invokes the utility. The following statements are examples of how to invoke the utility:

- EXEC PGM=IXCMINPD,PARM=' *notepadname* '
- EXEC PGM=IXCMINPD,PARM=' *notepadname* ,FORCE '

where *notepadname* specifies the name of the note pad to delete. The note pad name must adhere to the following rules:

- A note pad name can consist of two to four sections separated by periods.
- If a section is not specified, it is defaulted to all blanks.
- The first and the second sections must not be blank.
- Each section, if specified, must be left-justified with no trailing blanks.
- Each section can contain up to 8 uppercase alphanumeric (A-Z, 0-9), national (@, #, \$), or underscore () characters.

By default, the note pad is not deleted if it has active connectors. You can specify the optional FORCE keyword to force the deletion of a note pad even if it has connectors. The note pad is always deleted regardless of whether it has any notes or not.

You can delete XCF note pads only one at a time. To delete multiple note pads, you must run the IXCDELNP utility multiple times.

Sample JCL to run the IXCDELNP utility to delete an XCF note pad is shipped in SYS1.SAMPLIB member IXCDELNP.

Output messages from the IXCDELNP utility

The IXCDELNP utility issues one or more of the following messages.

Not authorized for update for MVSADMIN.XCF.IXCMINPD resource

SAF authorization checking for update access to MVSADMIN.XCF.IXCMINPD failed.

Input parameter not specified or too long

No PARM keyword was coded to pass the XCF note pad name as input, or the length of the PARM that was passed in was too large. For a description of the PARM keyword, see [Chapter 14, “Deletion utility for XCF note pads,” on page 403](#).

Utility input: *parm*

parm is the input that is specified with the PARM keyword.

Specified note pad delete option is not valid

The delete option that is specified with the PARM keyword is invalid. For a description of the PARM keyword, see [Chapter 14, “Deletion utility for XCF note pads,” on page 403](#).

Processing XCF note pad *notepadname*

The IXCDELNP utility successfully acquired the XCF note pad name *notepadname* from the input PARM keyword specification. The utility will attempt to delete the indicated note pad. Subsequent messages indicate the result of this delete processing.

XCF note pad deletion successful

The target note pad was successfully deleted.

XCF note pad name improperly specified

The note pad name that is specified with the PARM keyword is invalid. For a description of the PARM keyword, see [Chapter 14, “Deletion utility for XCF note pads,” on page 403](#).

Specified XCF note pad does not exist

The target note pad does not exist.

Specified XCF note pad not deleted. Note pad still in use by one or more connectors.

The target note pad was not deleted because it was still in use by one or more connectors. If necessary, you can force the deletion of a note pad that has active connectors by specifying the FORCE option on the PARM keyword. For a description of the PARM keyword, see [Chapter 14, “Deletion utility for XCF note pads,” on page 403](#).

No authority to note pad resources

The requester does not have the proper SAF authorization to delete the target note pad. See [“Authorizing XCF note pad requests” on page 147](#) for information about defining security profiles for note pad requests.

XCF Utility can not be invoked from the Master Address Space to delete a note pad

The MASTER address space cannot invoke the IXCDELNP utility. See [z/OS MVS Programming: Sysplex Services Guide](#) for information about authorization and environmental requirements for deleting a note pad.

Unexpected error. RC: *return_code* RSN: *reason_code* See IXCNOTE macro for return code and reason code explanations.

An unexpected return and reason code was returned from the IXCNOTE service. See [z/OS MVS Programming: Sysplex Services Reference](#) for IXCNOTE return and reason codes.

No security decision could be made

No security decision was made for the requester because there is either no security product installed on the system or no proper FACILITY class resource defined. See [“Authorizing XCF note pad requests” on page 147](#) for defining security profiles for note pad requests.

XCF Note Pad services not available on this system

XCF Note Pad services are not available on this system. The situation will persist for the life of the IPL.

The system is not configured to support access to the note pad catalog structure or note pad structure

The system is not currently configured to support note pads. See [Chapter 6, “Planning XCF Note Pad Services in a sysplex,” on page 141](#) for more information about setting up the XCF Note Pad Services.

Unable to obtain the system resources needed to process the request.

The IXCDELNP utility was unable to obtain the system resources that are needed to process the request. Retry the request after allowing time for the condition to clear.

The system does not have connectivity to the coupling facility that contains the note pad catalog structure.

The system does not have connectivity to the coupling facility that contains the note pad catalog structure. Try invoking the IXCDELNP utility on a different system that does have connectivity to the note pad catalog structure, or retry the request after the system regains connectivity.

The system does not have connectivity to the coupling facility that contains the note pad structure

The system does not have connectivity to the coupling facility that contains the note pad structure. Try invoking the IXCDELNP utility on a different system that does have connectivity to the note pad structure, or retry the request after the system regains connectivity.

A structure is temporarily unavailable due to system managed processing against the note pad catalog structure or note pad structure

The note pad catalog structure or the note pad structure is temporarily unavailable due to system managed processing. Retry the request after allowing time for the condition to clear.

The note pad is quiesced. Requests to process a note pad are rejected when a note pad is quiesced.

The note pad is quiesced, and the system was unable to process the delete request for the note pad. Retry the request after allowing time for the condition to clear.

XCF component error

The delete request failed due to an XCF component error.

Specified note pad name is too long

The note pad name that is specified with the PARM keyword is invalid. For a description of the PARM keyword, see [Chapter 14, “Deletion utility for XCF note pads,” on page 403](#).

Chapter 15. Calculating sizes for signaling services

This section contains detailed information for calculating the size of a coupling facility list structure for signaling and for calculating the size of the message buffers.

Calculating the size of a list structure for signaling

To calculate the size of a list structure for signaling, use the formula in *PR/SM Planning Guide*.

A list structure for signaling requires the following attributes:

- A list-structure-type that uses adjunct and data but neither keys nor names.
- An entry-to-element ratio of 1
- A list-element-characteristic value of 4
- A maximum data-list-entry size value of 16
- A lock-table-entry characteristic value of 0
- A lock-table-entry count value of 0
- The number of list headers required depends on the number of systems using the structure for pathin and/or pathout. Use the following formula to calculate the number of list headers:

```
#lists = 8 + #paths
    where:
        #paths =
            (#systems using the CF list structure as PATHOUT
             * #systems using the CF list structure as PATHIN) -
            #systems using the CF list structure as both
            PATHOUT and PATHIN
```

- List entries are tracked by entries (not by elements). XCF sets list limits on the individual lists in the structure.
- The structure disposition is DELETE.
- The connection disposition is DELETE.
- Structure rebuilds are allowed.
- Non-volatility is not required.
- List monitoring is used.

A list notification vector is allocated on each system to perform this monitoring. You need one vector for each signaling structure on each system in the sysplex that is using the structure. The number of entries in the vector is equal to the number of systems from which the system will receive signals through that signaling structure.

Additional considerations

The following topics provide more background on calculating the size of a signaling structure.

Required space

- z/OS requires eight list headers to manage a coupling facility list structure for signaling.
- So that at least one signaling path can be established, a list structure must be allocatable with at least one list header more than the number needed by MVS (#XCF_lists).
- To allow the maximum size signal supported by the MVS signaling service (61K bytes) to be inserted into a single list entry, a list structure used for signaling must be allocated so that a list entry can contain up to 64K bytes of data (up to sixteen 4096 byte data elements per data entry).

Therefore, in z/OS, a coupling facility list structure defined for signaling must have a minimum of twenty-five list entries. If this absolute minimum of 25 list entries is not available, the list structure remains defined to MVS but is considered inoperative and cannot be used for signaling.

Recommended space

The size specified for the coupling facility list structure in the CFRM policy should be large enough to allow for the recommended number of list headers and list entries, as described below. Specifying more than this minimum amount of space allows MVS to queue more signals on a list simultaneously and improves signaling performance.

- It is recommended that the coupling facility list structure be large enough to have at least one signal queued simultaneously on every list header used as a signaling path.

The number of data elements needed for these list entries depends on the size of the signals that are sent. For example, for a 61K byte signal (the largest signal supported by the MVS signaling service), 16 data elements would be needed. Since the entry to element ratio is 1:1, 16 list entries would also be allocated. Thus, to allow for one 61K byte signal to be queued simultaneously over every signaling path, the list structure must be large enough to accommodate 16 list entries per path.

- It is also desirable for the list structure to be large enough to accommodate the peak number of signals simultaneously queued over each list header for a signaling path.

Otherwise, MVS is forced to queue the signal on the sending system. This delays signal delivery and increases the amount of real storage needed for message buffers.

- MVS also needs list entries to manage the coupling facility list structure. The number and size of these list entries varies with:
 - The number of list headers in the list structure,
 - The maximum number of systems that can connect to the list structure,
 - The release of MVS that those systems are running.

In z/OS, it is recommended that the list structure be large enough to provide one list entry per connected system, plus eight additional list entries for control data. Each list entry requires only one data element.

If fewer than the recommended number of list entries are available, MVS might continue to use the coupling facility list structure for signaling, although possibly with some degradation.

Possible consequences of too little space

If either the coupling facility policy or the coupling facility list structure itself provides less than the recommended minimum amount of storage, signaling may be affected in undesirable ways. For example:

- If there is not enough space for one signal per signaling path, the potential for sympathy sickness exists because the failure of a system to process a signal could prevent other systems from transferring a signal. Thus, one non-operational system could interfere with the signal traffic between other unrelated systems, possibly causing a loss of signaling connectivity.
- A shortage of space (list entries) for a particular signaling path could cause signals to be queued on the sending system, thus delaying signal delivery and increasing the amount of real storage needed on the sending system for signal buffers.
- A shortage of space for control data could delay the establishment of signaling paths through the coupling facility list structure. Such delays could prevent an IPLing system from establishing signaling connectivity in a timely manner, or cause Sysplex Failure Management to partition systems out of the sysplex because signaling connectivity could not be reestablished quickly enough.

Vector space required on the MVS systems

The MVS signaling service needs to establish monitoring of every list in the coupling facility list structure (not every system must monitor every list though). A list notification vector is allocated on each system to perform this monitoring.

Basically you need one vector for each signaling structure on each system in the sysplex that is using that structure. The number of entries in the vector is equal to the number of systems from which the system will be receiving signals via that signaling structure.

If the vector cannot be allocated with enough capacity to support the minimum requirements for monitoring, the coupling facility list structure remains defined but cannot be used for signaling. The vector might satisfy the minimum requirements for monitoring, but might not be large enough to allow the system to monitor all the desired lists used for signaling paths. Signaling paths that cannot be established for this reason are considered inoperative.

When MVS detects circumstances in which successful allocation of the list notification vector might be possible, MVS automatically attempts to start using the coupling facility list structure for signaling again. If a vector was allocated, but was not large enough to support all the desired signaling paths, MVS attempts to increase the length of the vector. A rebuild of the coupling facility list structure is not a suitable recovery action since the resource problem is associated with the system itself.

The particular hardware configuration of the system, including the connectors to coupling facility list structures and their use of local vectors, will determine whether there is sufficient resource to allow allocation of the list notification vector. MVS cannot detect all the changes that might make it possible for the list transition vector to be allocated. Therefore, manual intervention may be required, either to make modifications to the configuration, or to cause the MVS signalling service to attempt to start using the coupling facility list structure for signaling again. When configuring their systems, customers should ensure that there is enough resource available to allow the needed list transition vectors to be established for each coupling facility list structure defined to the MVS signaling service.

Message information for some MVS groups

This topic provides information on the lengths, frequency, and performance requirements for messages sent by some MVS groups. This information can be useful when defining transport classes.

- Consoles

For the consoles XCF groups, SYSMCS and SYSMCS2, the length of messages vary. The maximum length of a message is 4K.

- DAE

The dump analysis elimination (SYSDAE) group issues one message per dump request, whether or not the dump is suppressed. Each message is approximately 400 bytes long.

- Global resource serialization

- RING mode

For the global resource serialization XCF group, SYSGRS, the most important message to consider is the RSA-message, which holds information about requests for global resources. The RSA-message can contain up to 32K bytes of data. The RSA-message will be sent frequently and will have high performance requirements. (For example, the performance of data base programs and the speed with which access to data sets is granted will depend on the speed with which the RSA-message is processed.) If an alternate serialization method is used, then the performance requirements for global resource serialization signaling are not as critical.

- STAR mode

Global resource serialization uses the SYSGRS group for QSCAN (GQSCAN and ISGQUERY), global ENQ contention notification, and Enhanced Contention Analysis (ECA) processing. Signals for these functions might be different in size. The maximum size a signal can be is 61K bytes. If more than 61K bytes of data is required to complete the request, the message is split into more than one part.

Global resource serialization also creates the SYSGRS2 XCF group in RING and STAR modes but does not use the SYSGRS2 XCF group in XCF communications. Global resource serialization is used in all sysplex configurations even if a different serialization product is used. Global requests that are managed by GRS must be processed immediately.

- JES2

The JES2 XCF group issues two kinds of messages:

- During a sysplex reconfiguration, messages of less than 1K in length will flow regularly between the JES2 XCF group members in the sysplex.
- As a JES2 system in a sysplex joins a network, it receives JES2 XCF messages with information about the other systems in the network. These messages can be up to 32K bytes in length and contain approximately 30 - 40 bytes of data about each system connected to the network. (Messages longer than 32K, are sent in two or more separate messages.)

- JES3

The JES3 group uses XCF messages to transport JES3 staging areas between the JES3 XCF group members in the sysplex. Most staging areas are less than 1K in length; however, the JES3 XCF group combines multiple staging areas and transports them in a single message that can be up to 64K bytes long.

- WLM

The WLM group, SYSWLM, sends a message approximately once every ten seconds. The length of the message depends on the number of service classes defined in the WLM policy definition. It is advantageous, but not critical, for the message to be received at once.

Estimate the size needed as follows: 320 bytes * the number of service class periods (in all service classes) that have a response time or velocity goal.

The message is sent by MVS images running in GOAL mode to other MVS images running in GOAL mode. If you have not yet migrated two systems to GOAL mode, no messages will be sent.

- XCF

The XCF group, SYSXCF, issues several different kinds of messages:

Signals requesting data from some other system in the sysplex

These signals are typically less than 956 bytes.

Signals responding to such requests

The length of these responding messages depends on the amount of data to be returned. As used by XCF, the response provides information about the members that reside on the system that is asked to gather the data. If the queried group on such system has only one member, the response is typically less than 956 bytes. For groups that have many members, tens of thousands of bytes of data is possible.

Signals issued by a system that voluntarily sends member data to other systems in the sysplex

Such signals are typically less than 956 bytes

Signals issued on behalf of an XCF Client/Server service exploiter

The sizes of these signals are based on the amount of data the exploiter wishes to send.

With current usage, the first three types of signals usually occur infrequently since they are sent in response to the DISPLAY,XCF,GROUP,grpname,memname command. They are also broadcast to the sysplex when an XCF group member is considered stalled. If the stall persists, the signals might reoccur periodically, typically on the order of minutes. If future applications elect to exploit the IXCMG GATHERFROM=OTHER service to collect data from other systems, the frequency, size, and performance requirements can change in response to their usage.

The frequency and size of the Client/Server related signals depend on the usage of the service exploiters.

Calculating message buffer space — An example

To limit the number of message buffers in your system and maintain adequate signaling performance, you need to know how much buffer space is required to support a particular message length. (The message buffer space required for a particular message length can vary from release to release.) For this release of MVS, compute the message buffer space required for a single message as follows: subtract 1K from

the message length, round the result up to the nearest multiple of 4K, then add 2K. For example, if the message length is 40,000 bytes:

```
40,000 - 1,024 = 38,976
(Rounding 38,976 up to the nearest multiple of 4K equals 40K)
40K + 2K = 42K
```

So, a MAXMSG value of 42K provides enough message buffer space to send a single message of 40,000 bytes. Because the transport class needs more than a single message buffer to have an adequate supply for signalling, the MAXMSG value, in this example, should be a multiple of 42K.

For details on deciding which MAXMSG value to use, see [Chapter 5, "Planning signaling services in a sysplex,"](#) on page 99.

Total Buffer Space: To compute the maximum message buffer space that MVS can allocate per system, use the following formula:

```
total = paths + classes + LOCALMSG value
```

```
Where:  paths          = the sum of all message buffer spaces
          specified for all paths.
        classes        = the sum of all message buffer
          spaces calculated for all transport classes.
        LOCALMSG value = the sum of local system message buffer
          spaces for all transport classes
```

To calculate the buffer space for a transport class, multiply 64K or the specified message buffer space (whichever is greater) by the number of systems in the sysplex.

For local message buffer spaces, add the LOCALMSG value for each transport class specified for the system.

For example, in a sysplex of four systems, suppose COUPLxx is defined as follows for one of the systems:

```
COUPLE  SYSPLEX(PLEXOF4)  MAXMSG(16)

CLASSDEF CLASS(DEFAULT)  MAXMSG(96)
CLASSDEF CLASS(TCGRS)    MAXMSG(32)

PATHIN  DEVICE(110,111)
PATHIN  DEVICE(112,113)  MAXMSG(48)

PATHOUT DEVICE(120,121)  CLASS(TCGRS)
PATHOUT DEVICE(8C40)    MAXMSG(50)
PATHOUT DEVICE(8C50)

LOCALMSG CLASS(DEFAULT)  MAXMSG(20)
```

For this system, the buffer space values for the paths and transport classes are:

Signal Path	Buffer Space	
-----	-----	
110	16K	(default from COUPLE MAXMSG)
111	16K	(default from COUPLE MAXMSG)
112	48K	(specified in PATHIN statement)
113	48K	(specified in PATHIN statement)
120	32K	(default from class TCGRS)
121	32K	(default from class TCGRS)
8C40	50K	(specified in PATHOUT statement)
8C50	96K	(default from class DEFAULT)
-----	-----	
Transport class	Buffer space	
-----	-----	
DEFAULT	404K	(96K x 4 systems, + 20K local increment specified for LOCALMSG)
TCGRS	256K	(Minimum 64K x 4 systems)

The total maximum buffer space for the system is:

```
Total = 16K + 16K + 48K + 48K + 32K + 32K + 50K + 96K + 404K + 256K
       = 998K
```

Calculating the size of list structures for system logger applications

Use the following procedure to calculate how large you should make each coupling facility list structure allocated for coupling facility log streams. This procedure applies only to coupling facility log streams. You can use this procedure for any system logger application, including applications provided by IBM, independent software vendors, or your own installation.

For logrec log stream and operations log log streams, specific sizing recommendations are provided in the *System/390 MVS Parallel Sysplex Configuration Volume 2: Cookbook*, SG24-2076.

For other IBM-provided system logger applications, such as CICS log manager, you should get recommendations for the size of your coupling facility structure or structures from documentation on that particular application.

The procedure will help you determine the minimum amount of coupling facility space you should allocate for a coupling facility structure, based on the amount of space needed for all the log streams mapping to the structure.

This procedure requires the use of the *PR/SM Planning Guide*, order number GA22-7123-13 or higher. Versions at a lower dash level than -13 are not valid with this sizing procedure. The example provided at the end of this procedure applies only for a coupling facility at the following levels:

- CFLEVEL 1 with a service level 4.03 or higher.
- CFLEVEL 2 or higher.

Go through the procedure once for every coupling facility structure.

1. Gather information about the coupling facility structure.

For each structure, gather the following information common for all the log streams that map to the structure. You will use these values in sizing the coupling facility space you need. Most of this information corresponds to parameters in the structure definition of the LOGR policy. You can specify the values you gather here when you add information to the LOGR policy (see [“Add information about log streams and coupling facility structures to the LOGR policy”](#) on page 280).

AVGBUFSIZE

Average size of log blocks written by applications to log streams associated with this coupling facility structure. This value corresponds to the AVGBUFSIZE parameter specified for each structure in the LOGR policy. For this procedure, pick the value that represents the average size of log blocks generated by all the log streams writing to this coupling facility structure. For information on specifying the AVGBUFSIZE parameter in the LOGR policy, see [“Specifying the average log block size”](#) on page 281.

For a logrec log stream, **IBM recommends** an AVGBUFSIZE of 4068.

For an OPERLOG log stream, **IBM recommends** an AVGBUFSIZE of 512.

For an RRS log stream, see [z/OS MVS Programming: Resource Recovery](#).

Installation or vendor supplied applications should base AVGBUFSIZE on the average size of the log blocks they want to write to the log stream.

MAXBUFSIZE

Maximum size of a log block corresponding to the MAXBUFSIZE parameter in the LOGR policy.

For a logrec or OPERLOG log stream, **IBM recommends** a MAXBUFSIZE of 4096.

For an RRS log stream, see [z/OS MVS Programming: Resource Recovery](#).

Installation or vendor supplied applications should base MAXBUFSIZE on the maximum size of the log blocks they want to write to the log stream.

Note that once you have defined the MAXBUFSIZE for a coupling facility structure, you cannot update the value. To change the MAXBUFSIZE, you must delete the log streams associated with the structure, delete the structure, and then re-define the structure with the new MAXGBUFSIZE value.

LOGSNUM

Maximum number of log streams in the structure corresponding to the LOGSNUM parameter in the LOGR policy. See [“The LOGSNUM parameter” on page 249](#) for more information.

MXSS-to-TSS

The ratio of the maximum structure size to the target structure size. For example, if you want your maximum structure size to be twice as big as your target structure size, then MXSS-to-TSS would be 2. See Appendix B in the *PR/SM Planning Guide*.

2. Gather information about each log stream.

For each log stream that maps to a coupling facility structure, gather the following information.

Most of these values are projected or desired values for your log stream, and a few correspond to parameters you will specify in the LOGR policy. You will use these values in sizing the coupling facility space you need.

LOWOFFLOAD

LOWOFFLOAD specifies the point, in percent value of space consumed, where system logger will stop offloading coupling facility log data to the DASD log data sets for this log stream. It corresponds to the LOWOFFLOAD parameter in the LOGR policy. For OPERLOG, logrec log stream, or the CICS log manager, **IBM recommends** that you use the LOWOFFLOAD default value of zero. See Chapter 12, [“Administrative data utility,” on page 335](#) for more information on the LOWOFFLOAD parameter.

HIGHOFFLOAD

HIGHOFFLOAD specifies the point, in percent value of space consumed, where system logger will begin offloading coupling facility log data to the DASD log data sets for this log stream. It corresponds to the HIGHOFFLOAD parameter in the LOGR policy.

For OPERLOG, logrec log stream, or the CICS log manager, **IBM recommends** that you use the HIGHOFFLOAD default value of 80.

See Chapter 12, [“Administrative data utility,” on page 335](#) for more information on the HIGHOFFLOAD parameter.

ResidencyTime

Desired residency time for log data in the coupling facility structure, in seconds. Residency time is the amount of time you want a log block to stay in the coupling facility between the time it is written to the coupling facility and being offloaded to DASD.

For a logrec or OPERLOG log stream, **IBM recommends** a residency of 10 seconds. For other system logger applications, get the desired residency time from the documentation for the application.

WritePerSec

Projected write requests per second against the log stream. This value is the total IXGWRITE requests per second issued by all system logger applications or systems connected to a log stream.

For an OPERLOG log stream, you can calculate projected writes per second using the current SYSLOGs. For each system in the sysplex, do the following calculation:

- a. Choose a spot in SYSLOG and note the timestamp at that spot. You might want to choose a peak or high usage time of day.
- b. Page down 2000 lines by issuing, for example, DOWN 2000 to SDSF or TSO/E browse.
- c. Note the time stamp in this spot.
- d. Calculate the number of seconds between the two time stamps. This is the number of seconds it took to write 2000 lines of SYSLOG. (This calculation is based on most messages being one line).
- e. Divide 2000 by the number of seconds it took to write 2000 lines to get the lines per second.
- f. Add the results for each system's SYSLOG to get the total writes per second for the OPERLOG log stream.

For a logrec log stream, calculate the projected writes per second using the current logrec data sets. For each system that will write to the logrec log stream, do the following calculation:

- Request or obtain an EREP report for a particular time span. It must be a report that includes all records. For example, take an EREP daily report that processed all records.
- Near the top of the report, message IFC120I will tell how many records were written in the time span chosen for the report. Divide this number by the number of seconds in the time span for the average writes per second for this logrec data set.
- You can also look at timestamps in the output to analyze how many records were written in a particular second. You can do this to check for peak usage.
- Add the results for each logrec data set that will write to the log stream to get the total writes per second for a logrec log stream.

3. Calculate coupling facility structure list entries required per log stream.

For each log stream that maps to the coupling facility structure, use the following formula to calculate the number of list structure entries required to represent a log stream's data. The number of entries a log stream needs in the coupling facility list structure depends on the coupling facility residency time desired for a given log stream. Use the values for WritePerSec and ResidencyTime gathered in the prior step:

```
Entries = ((100/(HighOffload-LowOffload))*WritePerSec*ResidencyTime)+4
```

4. Calculate the total coupling facility structure size.

This step calculates the size needed for each coupling facility structure to contain all the log data for the log streams that map to it. Perform this step for each coupling facility structure planned for system logger applications.

This step requires that you use the values from the previous steps and refer to Appendix B in *PR/SM Planning Guide* for formulas for the total structure size (TSS) and maximum structure size (MSS) for a coupling facility level 1 list structure at service level 4.01. This formula applies to the following coupling facility levels:

- CFLEVEL 1 with service level 4.03 or higher.
- CFLEVEL 2 or higher.

In order to pick the correct formula, you will need to know that the system logger uses keys and adjunct data, but no names for its list structures.

For each structure, use the following data as input to the TSS and MXSS formulas:

MLSEC

MLSEC is the number of coupling facility list entries needed for all the log streams that map to a coupling facility structure. Calculate this value by taking the value for **entries** calculated in step “3” on page 414 for the log stream with the largest **entries** value and do the following:

```
MLSEC = entries * maximum number concurrently  
         active log streams for the structure
```

You take the **entries** value for the largest log stream because when there are multiple log streams assigned to one coupling facility structure, system logger divides the structure space equally between each log stream connected to the structure.

If you are not sure how many of the log streams mapped to the structure will be concurrently active at any given time, multiply the entries times the LOGSNUM value for the structure to calculate the value for MLSEC. That way, you can be certain your coupling facility structure will be large enough for all logging activity.

MLSELC

Not used.

LELX

If MAXBUFSIZE is greater than 65,280, LELX = 1. Otherwise, LELX = 0.

MDLES

To calculate MDLES

```
(Greater of MAXBUFSIZE or 8192) / (256 * (2 ** LELX)).
```

LC

Do the following to calculate LC:

```
LOGSNUM * 4 + 1
```

LTEC

LTEC = 0

LTEX

LTEX = 0

R_le

R_le=1

R_data

Do the following to calculate R_data:

```
R_data=
(AVGBUFSIZE + 4 Bytes
rounded up to the next (256 * (2**LELX)) boundary)
/ (256*(2**LELX))
```

5. Calculate **SIZE** and **INITSIZE** parameter values for the structure definition in the CFRM policy.

Take the TSS and MXSS values for a level 1 coupling facility at service level 4.01 or higher and do the following to calculate SIZE and INITSIZE:

```
INITSIZE=TSS * 4
SIZE=MXSS * 4
```

Example of determining the size of a coupling facility structure.

The following example shows how to calculate the size of a coupling facility structure that has two log streams mapping to it. Note that this example was calculated for a level 1 coupling facility at service level 4.01 or higher.

- a. Gather information about the coupling facility structure.

AVGBUFSIZE

AVGBUFSIZE = 4068

MAXBUFSIZE

AVGBUFSIZE = 65,532

LOGSNUM

LOGSNUM = 2

MXSS-to-TSS

MXSS-to-TSS = 2

This means that the maximum structure size should be twice the initial structure size.

- b. Gather information about each log stream:

- Log stream information for log stream 1:

LOWOFFLOAD

LOWOFFLOAD = 0

HIGHOFFLOAD

HIGHOFFLOAD = 80

ResidencyTime

ResidencyTime = 10

Calculating Signaling Sizes

WritePerSec

WritePerSec = 5

- Log stream information for log stream 2:

LOWOFFLOAD

LOWOFFLOAD = 0

HIGHOFFLOAD

HIGHOFFLOAD = 80

ResidencyTime

ResidencyTime = 10

WritePerSec

WritePerSec = 20

- c. Calculate coupling facility structure list entries required per log stream.

For log stream 1:

```
Entries = ((100/(HighOffload-LowOffload))*WritePerSec*ResidencyTime)+4
Entries = ((100/(80-0))*5*10)+4
Entries = 67
```

For log stream 2:

```
Entries = ((100/(HighOffload-LowOffload))*WritePerSec*ResidencyTime)+4
Entries = ((100/(80-0))*20*10)+4
Entries = 254
```

- d. Calculate the total coupling facility structure size. Use the following data as input to the TSS and MXSS formulas in the *PR/SM Planning Guide*.

MLSEC

MLSEC = (254 * 2)

MLSEC = 508

Note that 254, the **entries** value for log stream 2 is used to calculate MLSEC because it is the larger of the entries values for the two log streams.

MLSEL

Not used.

LELX

LELX = 1

MDLES

MDLES = 128

LC

LC = 9

LTEC

LTEC = 0

LTEX

LTEX = 0

R_le

R_le = 1

R_data

R_data = 8

Calculate SIZE and INITSIZE parameter values for the structure definition in the CFRM policy from the TSS and MXSS values:

MXSS

MXSS = 1472

TSS

TSS = 704

```

SIZE = MXSS * 4
SIZE = 1472 * 4
SIZE = 5888

INITSIZE = TSS * 4
INITSIZE = 704 * 4
INITSIZE = 2816

```

6. **If you need to change the coupling facility structure size later**, there are two ways you can do that:

- Dynamically alter the coupling facility structure size by entering the SETXCF START,ALTER command or issuing the IXLALTER service.
- Update the CFRM policy with the new coupling facility size, activate the policy, and then have the operator initiate a rebuild of the structure. See [Chapter 12, “Administrative data utility,” on page 335](#) for updating the CFRM policy using the IXCMIAPU utility.

Note, however, that these options should not be substituted for advance planning, which can optimize both capacity and performance. See the following references for more information:

- [z/OS MVS System Commands](#) for the SETXCF command.
- [z/OS MVS Programming: Sysplex Services Reference](#) for reference information about the IXLALTER service.
- [z/OS MVS Programming: Sysplex Services Guide](#) for guidance on using the IXLALTER service.

Chapter 16. Sysplex configurations

IBM recommends that you provide redundant channels, signaling paths, Sysplex Timers, and shared couple data sets in order to avoid having a single point of failure in the sysplex.

For example, if a 3088 fails and there is not a second 3088 to maintain signaling connectivity between MVS systems in the sysplex, all but one system must be removed from the sysplex and put in a wait state. Or, if you do not have an alternate couple data set and the primary couple data set fails, all systems are put in a non-restartable wait state.

When a sysplex system is running under VM or in a logical partition that is sharing processor resources, you must ensure that its processor resources are sufficient to ensure good overall system performance. A system that is running without sufficient resources can degrade the performance of sysplex-wide communications functions, such as global resource serialization, signaling, and console communications.

- [“Under VM - Physical configuration of a sysplex with MVS Guests” on page 419](#)
- [“On PR/SM - Physical configuration of a sysplex on LPARs” on page 423](#)
- [“Configuration of a sysplex on LPAR and Non-LPAR systems” on page 424](#)

Under VM - Physical configuration of a sysplex with MVS Guests

z/VM Guest Coupling Simulation Support is the software that simulates the hardware and software required to run an MVS sysplex environment as second level guests under z/VM.

z/VM is able to simulate advanced Coupling Facility (CF) and Message Facility (MF) function, namely, the ability of a set of CFs to be directly connected to one another and to z/OS guests. z/VM Guest Coupling Simulation provides for the simulation of one or more complete parallel sysplexes within a single z/VM system image. The intent is to provide a pre-production testing platform for a coupled-system installation. The z/VM simulated environment is not intended for production use because its single points of failure negate the intent of the parallel sysplex environment. Other than the processors required, there is no special hardware needed: no coupling links and no external coupling facilities. Neither is such hardware supported. All guest operation systems coupling within a simulated sysplex can only be coupled (through simulated coupling links) to coupling facilities also running as guests of the same z/VM system. Up to 32 virtual machines can be coupled within a simulated sysplex, with each such virtual machine coupled to up to eight coupling facility virtual machines. In addition, when the processor and z/VM are so capable, each simulated coupling facility can connect to up to seven peer simulated coupling facilities. All coupled guests and simulated coupling facilities run within a single instance of the z/VM Control Program (CP).

See *z/VM Running Guest Operating Systems* for more information about setting up a sysplex environment, including setting up CFs under z/VM.

Components of the z/VM Guest Coupling Simulation Support

The z/VM Guest Coupling Simulation Support consists of three components: Coupling Facility Service Machines, Coupled Guests, and simulated Message Facility.

The CF service machine

This is a special type of z/VM service machine that runs the Coupling Facility Control Code (CFCC). This code is not part of z/VM and is loaded directly from the processor. Because z/VM is executing the actual code that runs in a real coupling facility, there is basically no difference in function between a real coupling facility and z/VM's implementation.

The system administrator must define a user ID for each CF Service Machine that will run under z/VM. To define each user ID as a CF Service Machine, the CFVM operand must be specified on the OPTION directory control statement of each CF Service Machine. The CFVM operand defines the virtual machine as a CF Service Machine and allow connections between itself and coupled guests.

You can define up to eight CF Service Machines. Up to 32 coupled guests can be connected to each CF Service Machine. When the processor and z/VM are so capable, each CF Service Machine can couple to up to seven other CF Service Machines.

**Attention:**

Make sure to provide the coupling facility service machines with sufficient CPU resources. Otherwise, the coupled guests might experience spin loop timeout or loss of connectivity to the CF.

The coupled guest

This is a uni-processor (UP) or multi-processor (MP) virtual machine that has a coupled connection to a CF Service Machine using the software simulation of the message facility. This virtual machine is running an operating system that supports the coupling facility, such as MVS.

The system administrator must define a user ID for each coupled guest that will run under z/VM. To define each user ID as a coupled guest, the CFUSER operand must be specified on the OPTION directory control statement for each virtual machine. The CFUSER operand defines the virtual machine as a coupled guest and allows it to connect to a CF Service Machine.

Any number of coupled guests can be defined, and each coupled guest can establish a coupled connection with up to eight different CF Service Machines concurrently.

The message facility simulation

The Message Facility, which is an optional part of the channel subsystem, is the hardware that connects a processor complex to the coupling facility. CP simulates the message facility for the benefit of CF Service Machines and coupled guests. The simulated Message Facility allows a coupled guest to exchange information with a CF Service Machine, and when the hardware and z/VM are so capable, it allows CF Service Machines to exchange information with each other. From the coupled guest's point of view, the Message Facility appears as a set of virtual message devices. The CF Service Machine's view of the Message Facility is proprietary.

Setting up a guest coupling environment under z/VM

The following sections describe how to set up a guest coupling environment under z/VM.

Define the z/VM service machines to MVS

Before MVS can use the z/VM Guest Coupling Simulation support, the administrative policy data in the Coupling Facility Resource Manager (CFRM) policy must be updated to define the CF Service Machines to MVS. [Figure 79 on page 421](#) is an example of the JCL required to define a sysplex. In this example, the sysplex will use two z/VM CF Service Machines. The user ID of the first CF Service Machine is CFCC1 and the user ID of the second is CFCC2.

```

//UPADJOB JOB
//*****
//*
//* EXAMPLE JCL TO UPDATE THE ADMINISTRATIVE POLICY DATA IN THE *
//* COUPLE DATA SET FOR CFRM (COUPLING FACILITY RESOURCE MANAGER) *
//*
//*****
//STEP20 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSIN DD *
DATA TYPE(CFRM) REPORT(YES)

DEFINE POLICY NAME(POLICY1) REPLACE(YES)

CF NAME(CFCC1
TYPE(SIMDEV)
MFG(IBM)
PLANT(EN)
SEQUENCE(0000000CFCC1)
PARTITION(0)
CPCID(00)
DUMPSPACE(2000)

CF NAME(CFCC2
TYPE(SIMDEV)
MFG(IBM)
PLANT(EN)
SEQUENCE(0000000CFCC2)
PARTITION(0)
CPCID(00)
DUMPSPACE(2000)

STRUCTURE NAME(LIST_01)
SIZE(10000)
INITSIZE(1000)
PREFLIST(CFCC1,CFCC2)
EXCLLIST(CACHE_01)

STRUCTURE NAME(CACHE_01)
SIZE(1000)
REBUILDPERCENT(25)
PREFLIST(CFCC2,CFCC1)
EXCLLIST(LIST_01)

```

Figure 79. Example: JCL to define the CF Service Machines to MVS

The information for TYPE(), MFG(), PLANT(), and SEQUENCE() can be obtained by using the CP QUERY MSGPROC command. The TYPE(), MFG(), and PLANT() information will always be the same when using z/VM Guest Coupling Simulation support. The SEQUENCE() value must be unique and is based on the user ID of the CF Service Machine.

Establish data sharing among MVS virtual machines

Working allegiance simulation must be used when running two or more MVS guests as part of a sysplex configuration using z/VM Guest Coupling Simulation support. The WRKALLEG operand on the MINIOPT directory statement or the CP SET WRKALLEG command must be used for any minidisk containing CFRM data sets to maintain cross-system lock integrity (and thereby, data integrity) within the sysplex.

Modify the MVS clock definitions

z/VM Guest Coupling simulation support does not support or simulate Sysplex Timers (ETR). When running in a z/VM environment, all virtual machine Time-of-Day (TOD) clocks are synchronized with the system TOD clock. This implicit synchronization allows all MVS images running in a sysplex on z/VM to have identically set TOD clocks without the use of a Sysplex Timer. To inform MVS that clock synchronization will be done by z/VM, you must either modify the CLOCK00 parmlib member or create a new CLOCKxx member for running under z/VM. Otherwise, you will not be able to run in sysplex mode with multiple MVS images. Only Monoplex mode will be supported.

Figure 80 on page 422 shows a sample CLOCKxx member required to run a sysplex under z/VM.

```

TIMEZONE W.04.00.00
ETRMODE YES
ETRDELTA 10
ETRZONE YES
SIMETRID 00

```

Figure 80. Sample: definition of CLOCKxx member for z/VM

Change and synchronize the time-of-day Clocks

z/VM allows you to simulate running your sysplex environment with a date and time that is different from the system date and time with the CP SET VTOD command.

In a z/VM environment, all virtual machine Time-of-Day (TOD) clocks are synchronized with the system TOD clock when they log on. If you want to run the virtual machines in a sysplex with a TOD setting that is different from the current system TOD setting, you must change and synchronize all the virtual machine TOD clocks within the sysplex.

To change the TOD clocks for the coupled guests, prior to starting your sysplex issue the following sequence of commands:

1. From the CGUEST1 user ID, issue the DEFINE MSGPROC commands to add CFCC1 and CFCC2 to your I/O configuration. Because CGUEST1 has only Class G privileges, the SET VTOD command in the next step requires that the CF Service Machine whose date and time that you want to set be in the I/O configuration of the coupled guest that is issuing the SET VTOD command.

```
def msgp cfcc1 vdev 400
```

```
def msgp cfcc2 vdev 500
```

2. From the CGUEST1 user ID, issue the following commands, substituting the appropriate values for the date and time.

Set the TOD clock:

```
set vtod date mm/dd/yyyy time hh:mm:ss
```

Set the TOD clocks for the CF Service Machines:

```
set vtod cfcc1
```

```
set vtod cfcc2
```

3. Issue the following command from the CGUEST2 user ID to synchronize the CGUEST2 TOD clock with the sysplex:

```
set vtod fromuser cfcc1
```

4. Issue the following command from the CGUEST3 user ID to synchronize the CGUEST3 TOD clock with the sysplex:

```
set vtod fromuser cfcc1
```

See *z/VM: CP Commands and Utilities Reference* for a complete description of these commands.

Understanding the node descriptor

Because z/VM Guest Coupling Simulation Support simulates a real coupling facility, the software must create a unique Node Descriptor (ND) for every CF Service Machine to be used. z/VM dynamically creates an architecturally correct unique ND when the message facility environment is created in the CF Service Machine. The method used to create the DN is documented so that the user can generate the

administrative policy for CFRM in MVS. This will allow MVS's CFRM policy generator to be used and there is no need for the user to specify an ND. This is exactly how it works with the real hardware.

Use the following information when defining a coupling facility in the CFRM Administrative Policy Utility, IXCMIAPU:

- Type Number (TYPE) - SIMDEV specifies that this is a z/VM simulated message device type.
- Manufacturer (MFG) - IBM indicates IBM as the manufacturer.
- Plant (PLANT) - EN indicates Endicott as the manufacturing plant.
- Sequence # (SEQUENCE) - The 12-byte field will contain the CF Service Machine user ID right justified. Pad all leading unused characters with EBCDIC zeros. For example, if a CF Service Machine of CFCC1 is specified, SEQUENCE is 0000000CFCC1.

On PR/SM - Physical configuration of a sysplex on LPARs

Figure 81 on page 423 shows the physical connections needed to connect two MVS systems on two logical partitions (LPARs) in one PR/SM processor and provide redundant channels, signaling paths, and couple data sets.

In this configuration, a failure on any one channel, path, ESCD, or couple data set does not bring down the sysplex. In the event of a failure, MVS can use a different channel, path, ESCD, or alternate couple data set to maintain sysplex operation.

- The figure shows channel to channel connections through ESCDs.
- A channel on each LPAR is connected to the primary couple data set and a second channel is connected to the alternate couple data set; both data sets are shared by the two LPARs.
- A Sysplex Timer is not needed in this sysplex because MVS can use the clock of the processor to synchronize timestamps. However, in this sysplex, a Sysplex Timer could be used.
- A coupling facility could be used in this configuration.

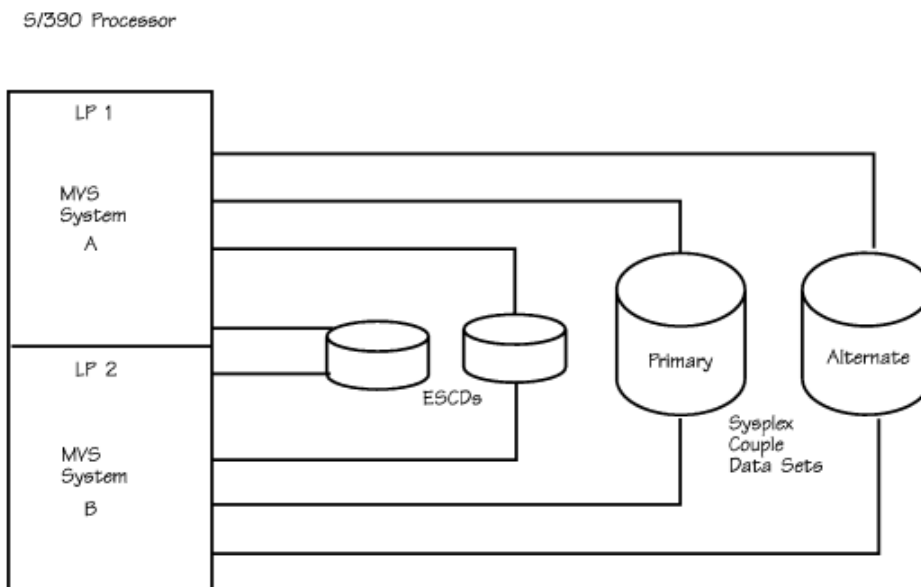


Figure 81. On PR/SM - Physical Configuration of a Sysplex on LPARs

Note: If you have PR/SM installed with the ESCON Multiple Image Facility (EMIF), you can share ESCON channels among logical partitions. For information about EMIF and shared channel paths, see *PR/SM Planning Guide* and [z/OS HCD Planning](#).

Configuration of a sysplex on LPAR and Non-LPAR systems

Figure 82 on page 424 shows the connections required to connect two processors, one of which runs two MVS images on PR/SM. It also illustrates redundant sysplex couple data sets, channels, signaling paths, and Sysplex Timers.

In this configuration, a failure on any one channel, path, ESCD, or sysplex couple data set does not bring down the sysplex. In case of a failure, MVS can use a different sysplex couple data set, channel, path, ESCD, or Sysplex Timer to maintain sysplex operation.

- The figure shows channel to channel connections through ESCDs.
- A channel on each LPAR in CPC1 is connected to the primary couple data set and a second channel is connected to the alternate couple data set. A channel on CPC2 is connected to the primary couple data set and a second channel is connected to the alternate couple data set. Both data sets are shared by all three systems.
- A Sysplex Timer is needed to synchronize timestamps between the two processors.
- A coupling facility can be used in this configuration.

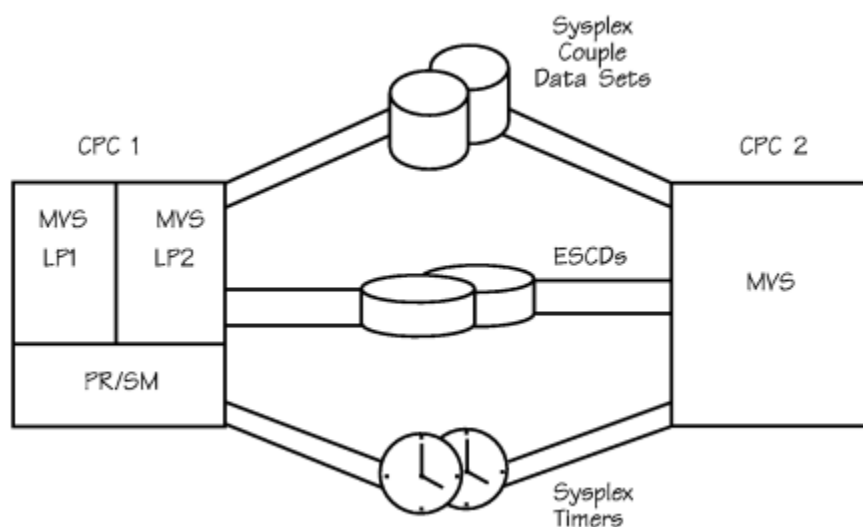


Figure 82. Configuration of a Sysplex on LPAR and Non-LPAR Systems

Note: If you have PR/SM installed with the ESCON Multiple Image Facility (EMIF), you can share ESCON channels among logical partitions. For information about EMIF and shared channel paths, see *PR/SM Planning Guide* and [z/OS HCD Planning](#).

Chapter 17. Coupling Facility Guidelines

There may be occasions when it is necessary to alter the Parallel Sysplex configuration — to add or modify processors, to upgrade coupling facilities, or to remove and replace hardware or software components of the configuration. This topic provides guidelines specifically for handling the configuration of coupling facility (CF) resources in a Parallel Sysplex environment. It emphasizes the relationship between the structures that may already be allocated in a CF and the CF and structure definitions in a coupling facility resource management (CFRM) policy. It is important to understand these relationships thoroughly before beginning any configuration changes for CF resources.

The following topics are included in this section:

- “A Review of Some Basic Concepts” on page 425.
- “Guidelines for coupling facility reconfiguration” on page 427.
- “Guidelines for unplanned outages” on page 449.

A Review of Some Basic Concepts

This topic reviews the concept of persistence and the role of CFRM in a Parallel Sysplex environment. Understand this topic thoroughly before undergoing CF reconfiguration.

A Review of the Use of Structure Data

An application instance using a CF structure through an active CF structure connection is referred to as a connector. The data stored in a CF structure as well as the use of the data is application-dependent. Connectors can store data in a structure that, in the event of a connector failure, can be recovered by a new (restarted) instance of the application. Alternately, connectors can store data in a structure that allows another connector (from another application instance) to perform recovery in the event of a failure. The concept of a bond between the data in the structure and a connector to the structure is referred to as a recovery bind. Once a connector has performed the recovery processing required by the application, XCF removes the recovery bind. For example, when an instance of IRLM fails, the failed IRLM has a recovery bind to data in the structure—in this case, information about retained locks. The information about retained locks is required for IRLM recovery processing.

A Review of the Concept of Persistence

Connection persistence implies the existence of a recovery bind between a connector to a structure and some data that is stored in the structure. For example, when an instance of IRLM fails, the failed connection becomes a failed-persistent connection. This indicates that the instance of IRLM has a recovery bind to data in the structure, which should persist even after the connector has terminated. A failed-persistent connection has a recovery bind that will be removed after recovery is performed by the application. It might be necessary to restart the application to initiate recovery actions that will remove the recovery bind. Non-persistent connectors, on the other hand, have no such recovery binds.

Structure persistence implies that the structure remains allocated in the CF even after there are no connections (active or failed-persistent) to the structure. Generally speaking, applications use a persistent structure for permanent data — data that needs to remain in a CF structure even if there are no connections to that structure. A persistent structure is not deleted when all connections to the structure cease to exist.

Non-persistent structures are generally used by applications with transient data, or data that must persist only when it is actively being used by connectors, or when there is a failed-persistent connection. A non-persistent structure is deleted when all connections to the structure cease to exist.

Note that a failed-persistent connector is bound to a particular allocated instance of the structure; MVS will not deallocate that instance of the structure, regardless of structure persistence or non-persistence, as long as any failed-persistent connector to the structure remains in existence.

Specifying Structure and Connection Persistence

Connector and structure persistence are for the most part independent choices that the application using a structure makes based on its design philosophy and considerations. For example:

- An application that stored permanent data in the CF, but that did not have a concept of recovery binds between data in the structure and any particular connector, would probably use a persistent structure with non-persistent connections.
- An application that stored transient data in the CF, but that needed some connector-related data in the structure to persist for recovery purposes in the event of a connector failure, would probably use a non-persistent structure with persistent connections.

Sample Persistence Scenarios

The following examples describe the effect of persistence in a sysplex.

Persistent Scenarios from a Connector's View

The behavior of persistent connections and structures is fairly straightforward in scenarios where it is the connections to the structures that come and go over time. The following general rules apply to a connector's actions:

Normal disconnect

When a persistent connector to a structure disconnects normally, it does not become a failed-persistent connections; it becomes *undefined*. If no other active or failed-persistent connections to the structure remain at the time of the disconnect, the structure is deallocated (if the structure is non-persistent) or remains allocated (if the structure is persistent).

Abnormal disconnect

When a persistent connector to a structure disconnects abnormally or terminates abnormally, indicating that recovery should be done at some point in the future, the connection is placed into the failed-persistent state. The structure will not be deallocated while this failed-persistent connection exists, regardless of whether the structure itself is persistent or non-persistent.

Reconnect

When an application associated with a failed-persistent connection restarts and reconnects to the structure, the connector will perform the recovery processing that was associated with its recovery bind to the structure (the details of this recovery processing are, of course, specific to the application). It will then resume normal use of the structure as an active connector.

Persistent Scenarios in Coupling Facility Failure Situations

When a CF fails, the scenarios involving persistent connectors to structures in the CF are a little more complicated and perhaps less intuitive. Generally, the following occurs:

- When the CF fails, a loss of coupling facility connectivity is recognized by attached z/OS systems. If a structure is duplexed, duplexing will be stopped to allow the structure instance in the other CF to be used without further recovery actions. If a structure is not duplexed, the loss of CF connectivity is reported to the connectors to the structure.
- If the CF structure connectors are able to rebuild the structure (into a different CF), the connections remain active and switch over to using the new rebuilt structure instance.
- If rebuild is not supported or fails, the connectors disconnect abnormally, indicating a recovery bind to the structure for which connectivity was lost. The persistent connections enter the disconnecting/failing state, and when the failures have been seen and confirmed by other connectors to the structure, the persistent connections enter the failed-persistent state.
- When an application associated with a failed-persistent connection attempts to reconnect to the structure, because there is still a recovery bind to the old instance of the structure (which, as far as

z/OS knows, is still allocated in the CF to which connectivity was lost), z/OS will attempt to reconnect to that old structure instance. There are several cases here:

- If the CF has been restored and the structure is still allocated in that coupling facility (that is, there was only a loss of connectivity to the CF, as opposed to a loss of the coupling facility and its data), then the reconnect will be successful and the connector will do recovery processing using the data in the structure. Generally speaking, recovery from logs or other manual recovery procedures will not be necessary in this case.
- If the CF has been restored and the structure is no longer allocated in that CF (that is, the CF failed and the structure and all of its data was lost), z/OS realizes that the old instance of the structure is no longer allocated, allocates a new instance of the structure, and reports to the connector that the connection request succeeded but resulted in the allocation of a new structure instance. The application will need to accomplish its recovery through alternative means, because the data in the structure is not present in this case. This might mean recovery from logs or other manual recovery procedures that must be performed.
- If the CF is still unavailable to the system attempting to reconnect but available to some system in the Parallel Sysplex, the attempt to connect to the structure will fail for that reason. If the CF remains unavailable to the system indefinitely, then attempts to connect to the structure will continue to fail indefinitely.

Connectivity should be restored between the system and the CF. If this cannot be done, it might be possible to restart the application on a system that has connectivity to the CF, or to manually rebuild the structure into a CF that has the required connectivity. As the last resort, when no active connectors to the structure remain, deallocation of the structure can be manually forced - allowing an attempt to connect to the structure to allocate a new structure instance as in the previous bullet.

- If the CF is still unavailable to all systems in the sysplex, CFRM will force deallocation of the inaccessible structure instance. Since no system has connectivity to the CF, the structure instance will become pending deletion. The attempt to reconnect will cause a new instance of the structure to be allocated in a different CF. The application will need to accomplish its recovery through alternative means, because the data in the structure is not present in this case. This might mean recovery from logs or other manual recovery procedures must be performed.

For more information on structure and connection persistence, and on forcing persistent structures and connections, see *z/OS MVS Programming: Sysplex Services Guide*.

Guidelines for coupling facility reconfiguration

This topic assumes that CFRM is in use in the sysplex. For configurations that are not yet using CFRM, see "Considerations for Policy Couple Data Sets" in Chapter 3, "Planning the couple data sets," on page 27 for more information on defining a CFRM couple data set to z/OS in order to use CFRM in the sysplex. Chapter 4, "Managing coupling facility resources," on page 41 provides detailed information about CFRM and how it maintains status for coupling facilities and structures in a Parallel Sysplex configuration. Remember the following important points when changing a CF configuration:

- CFRM uses the CFRM couple data set (CDS) to maintain status data about the current use of CF resources in the sysplex. This status data is referred to as the CFRM active policy (or just *active policy*).
- Activating a CFRM policy initiates a transition to the policy that might involve various changes to the CFRM active policy. The changes might remain pending if CF resources are in use. Operator intervention might be required to allow CFRM to make the pending policy changes. Consider the following conditions:
 - A CFRM policy contains CF definitions that associate a name with a CF (identified by a hardware node descriptor, partition, and CPCID). If a CF being used by the sysplex has a structure allocated in it, activating a policy that does not define the CF causes the CF definition to become pending deletion from the CFRM active policy. The CF will not be deleted from the active policy until all structures are removed from the CF.
 - If a structure is allocated, activating a CFRM policy that does not define the structure causes the structure definition to become pending deletion from the CFRM active policy. The structure will not

be deleted from the active policy until all instances of the structure are removed from coupling facilities in the active policy - this includes discontinuing use of the structure by the application.

- If a structure is allocated, activating a CFRM policy that would cause a change to the structure definition in the CFRM active policy may become pending. REALLOCATE or structure rebuild is needed to make pending structure changes in the active policy when the change cannot be made immediately.

This topic provides guidelines and procedures for the following tasks:

- Adding/defining a CF to the configuration
- Removing/deleting a CF from the configuration
- CF maintenance
- CF hardware reconfiguration

For guidelines and a procedure to add or define a CF to the configuration, see [“Adding a Coupling Facility” on page 429](#).

For guidelines and procedures to remove or delete a CF from the configuration, see the following topics:

- [“Planning for a Coupling Facility Shutdown” on page 430](#)
- [“Removing a coupling facility from the configuration” on page 432](#)

If all coupling facilities are to be deleted from the configuration, see [“Transitioning from a Parallel Sysplex to a base sysplex” on page 436](#) instead.

For guidelines and procedures to define or delete CF structures, see the following topics:

- [“Adding a structure” on page 430](#)
- [“Deleting a structure from the CFRM active policy” on page 440](#)

If all structures are to be deleted from the configuration, see [“Transitioning from a Parallel Sysplex to a base sysplex” on page 436](#) instead.

For guidelines and a procedure for CF maintenance, see the following topics:

- [“Planning for a Coupling Facility Shutdown” on page 430](#)
- [“Shutting down a coupling facility for maintenance” on page 431](#)

CF hardware reconfiguration includes the following tasks:

- Replacing one CF with another CF
- Moving a CF from one CPC to another CPC
- Moving a CF from one PR/SM partition to another on the same CPC.

See [“Planning for a Coupling Facility Shutdown” on page 430](#) before performing CF hardware reconfiguration. There are multiple options for CF hardware reconfiguration:

- When the reconfiguration involves replacing or moving all coupling facilities at the same time, follow instructions on [“Upgrading or replacing all coupling facilities” on page 434](#).
- When multiple coupling facilities exist, follow the procedures defined in [“Upgrading or replacing a coupling facility” on page 433](#).

This might involve changing the I/O configuration to delete and re-add the CF from the I/O configuration. This will ensure configuration-sensitive products are synchronized with the reconfiguration.

Alternatively, based on the needs of the installation, the reconfiguration can be accomplished without affecting the I/O configuration. The existing I/O configuration can be "reused" as long as CHPID information for the CF has not changed as defined for the z/OS images. Similarly, the SNA address used for the new CF must be the same as the address used for the old CF.

- Alternatively, follow the procedures for [“Removing a coupling facility from the configuration” on page 432](#), including deleting its definition from the I/O configuration for the affected z/OS image(s). Then

follow the procedures for “Adding a Coupling Facility” on page 429 to add the CF to the sysplex. This will ensure that sensitive products are synchronized with the reconfiguration.

For additional information, see the following publications:

- [z/OS MVS System Commands](#)
- [z/OS HCD Planning](#)
- [z/OS HCD User's Guide](#)
- [PR/SM Planning Guide](#)
- IOCP User's Guide

Adding a Coupling Facility

This topic has a procedure to add a coupling facility (CF) to an active sysplex.

1. Use HCD to create an IODF that defines the CF to the I/O configuration. See [z/OS HCD Planning](#) and [z/OS HCD User's Guide](#) for more information on defining coupling facility elements in a sysplex.
 - a. Create a new IODF containing the new CF control unit definition and channel path definitions that will be used to connect the CF to other images (z/OS images as well as CF images).
 - b. Use the HCD Build IOCDS function to create and download an IOCDS containing the CF channel path definitions for remote IOCDS management.

2. Activate the new IODF on all images (both z/OS and CF).

The IODF can be dynamically activated on z/OS systems. To ensure that the required processing for changes to CF elements (CF control units or CF channel paths) will be executed, either specify SOFT=VALIDATE on the ACTIVATE system command, or choose the "Activate software configuration only and validate hardware changes" option from the HCD Activate New Configuration panel. SOFT=VALIDATE is a requirement in all N-1 partitions when changes to CF elements are made.

Verify the addition of the CF on all systems with the following system commands:

```
DISPLAY CF
DISPLAY M=CHP(chpxx,chpyy...)
```

3. Ensure that any necessary hardware is installed (for example, ISC or ICB links) to provide connectivity between the z/OS image(s) and the CF. Connectivity must also be provided between coupling facilities that will be used for system-managed CF structure duplexing.
4. Ensure that CF CHPIDs are online. The following CF command can be used on a CF to configure a CF CHPID online:

```
CONFIGURE CHPID ONLINE
```

CHPIDs in use can be verified with the following CF commands:

```
DISPLAY CHPIDS
DISPLAY RESOURCES
```

See [PR/SM Planning Guide](#) for more information on coupling facility control code commands.

5. Create a new CFRM policy (or update an existing CFRM policy) that has a CF definition statement for the new CF to be added, and has the CF name added to all appropriate structure preference lists. See “Planning a coupling facility policy” on page 49 for more information. Activate the new CFRM policy using the SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname2 system command.

The CF should be added immediately to the CFRM active policy, but the addition of the CF to the preference list of allocated structures may remain pending until REALLOCATE or rebuild can make the change.

The addition of the CF to the active policy can be verified with the DISPLAY XCF,CF,CFNAME=cfname system command.

6. On all z/OS systems, configure CHPIDs online using the `CONFIG CHPID(chpxx, chpyy, . . .), ONLINE` system command.

The availability of the CF can be verified on all systems with the following system commands:

```
DISPLAY XCF, CF, CFNAME=cfname
DISPLAY M=CHP(chpxx, chpyy. . .)
DISPLAY CF
```

7. The new CF will not contain any structures. See "Using `REALLOCATE` or `POPULATECF` to relocate coupling facility structures" in Chapter 4, "Managing coupling facility resources," on page 41 for a description of how to move structures to their most preferred CFs after a CF has been added to the configuration.

Adding a structure

Create a CFRM policy (or update an existing CFRM policy) that has a structure definition for the structure to be added. For more information, see "Planning a coupling facility policy" on page 49. Activate the CFRM policy by using the system command `SETXCF START, POLICY, TYPE=CFRM, POLNAME=polname`.

Add the new structure immediately to the CFRM active policy. See application-specific instructions on how to initiate use of the new structure definition.

Planning for a Coupling Facility Shutdown

Before removing a CF from a Parallel Sysplex configuration, whether for replacement or maintenance, consider the following:

- Remove only one CF at a time from service. Perform maintenance for the CF during off-peak periods. Ensure structure preference lists in the CFRM active policy contain any alternate CFs to be used for structure rebuild (or duplexing).
- Most configurations should have multiple coupling facilities installed for improved availability. Ensure that all systems currently using a structure in the CF to be removed have connectivity to the alternate CFs.
- If the CF being removed contains an XCF signaling structure, ensure that a full set of redundant signaling paths are in place. For example, if a signaling connection consists of only a single signaling structure (and no CTC connections), signaling connectivity is lost during the structure rebuild process. The loss of signaling connectivity lengthens the time it takes to rebuild the signaling structure and might result in XCF timeouts,

XCF signaling structures should be rebuilt one at a time and not concurrent with any other structure rebuild. Note that `REALLOCATE` processing will rebuild structures one at a time.

If redundant signaling paths for all systems are not available during the maintenance procedure, consider increasing the following values in to prevent an XCF timeout:

- Use the `SETXCF COUPLE` command to change the system failure detection interval (`INTERVAL`). It might be necessary to also change this value in the `SYS1.PARMLIB` member `COUPLExx`.
- In a global resource serialization ring complex, use the `SETGRS` command to change the maximum tolerance time interval global resource serialization allows the RSA-message to return to a system (`TOLINT`). It may also be necessary to change this value in the `SYS1.PARMLIB` member `GRSCNFxx`.

If redundant signaling paths between all pairs of systems are not available during the maintenance procedure, it is recommended that SFM actions to remove a system for signaling connectivity failures be disabled, either by disabling SFM or by temporarily changing the SFM policy to specify `CONNFAIL(NO)` during the maintenance period.

- Ensure that enough capacity (storage, CP cycles, link capacity, and structure IDs) exists on the alternate CFs.

If there is not enough capacity on the alternate CFs to accommodate all structures to be rebuilt, consider deleting structures (see [“Deleting structures”](#) on page 440). In particular, consider deleting the following structures:

- For XCF signaling structures, use CTC connections for XCF signaling and stop XCF signaling paths through the CF structures. See [“Deleting the XCF signaling structure”](#) on page 444.
- For JES2 structures, place the checkpoint on DASD and remove the CF structure. See [“Deleting the JES2 checkpoint structure”](#) on page 440.
- For RACF structures, use RACF in non-data sharing mode, which will automatically delete the RACF structures. See [“Deleting the RACF database cache structures”](#) on page 442.
- When using Server Time Protocol (STP) to maintain time synchronization, consider the effect of the reconfiguration on the Coordinated Timing Network (CTN). See the *Server Time Protocol Implementation Guide* redbook for more information.

Shutting down a coupling facility for maintenance

For additional information, see [Best Practices Upgrading a Coupling Facility Version 2](#) (www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101905).

This topic provides a procedure for shutting down a coupling facility (CF) for maintenance when all CF structures can be removed from the CF and hardware identification information for the named CF is not changing. If not all structures can be removed from the CF (for example, the Parallel Sysplex configuration is a global resource serialization star complex with a single usable CF), then CF maintenance procedures should be performed while all systems are inactive. If the CF hardware identification will change, see [“Upgrading or replacing a coupling facility”](#) on page 433 instead.

The procedures in this topic require that all systems are at z/OS V1R9.0 or above in order to exploit maintenance mode. If this requirement is not met, the CF must be deleted and then re-added. See [“Removing a coupling facility from the configuration”](#) on page 432.

The procedures in this topic require that the CF contains no structures in transition or pending deallocation. If the CF contains a structure in transition or pending deallocation, connectivity to the CF should be restored. If connectivity to the CF cannot be restored, the CF must be deleted and then re-added. See [“Removing a coupling facility from the configuration”](#) on page 432. Structures that are in transition or pending deallocation are marked as (TRN) or (PND) in the output of the system command `DISPLAY XCF,CF,CFNAME=cfname`.

To exploit maintenance mode for shutting down a CF, consider the following procedure:

1. Prevent systems from allocating any structures in the CF by placing the CF in maintenance mode using the `SETXCF START,MAINTMODE,CFNAME=cfname` command.

Verify the denial of structure allocation due to maintenance mode with the `DISPLAY XCF,CF,CFNAME=cfname` command.

2. Remove all structures from the CF. See [“Removing structures from a coupling facility for shutdown”](#) on page 437 for guidelines on removing all structures from a CF that is in maintenance mode.

Verify the removal of all structures from the CF with the `DISPLAY XCF,CF,CFNAME=cfname` command.

3. On all systems, place all paths to the CF offline with the `VARY PATH(cfname,chpxx,cfname,chpyy...),OFFLINE,UNCOND` command.

The logical removal of all system connections to the CF should be verified with the following system commands:

```
DISPLAY XCF,CF,CFNAME=cfname
ROUTE *ALL,DISPLAY CF,CFNAME=cfname
```

4. SHUTDOWN the CF.

CF maintenance can be performed when preceding actions are complete.

If making changes to CF elements (CF control units or CF channel paths) in the I/O configuration, ensure that `SOFT=VALIDATE` is specified on the `ACTIVATE` system command. `SOFT=VALIDATE` is a requirement in all N-1 partitions when changes to CF elements are made.

Use the following procedure to bring the CF back into use after maintenance is complete:

1. When CF maintenance is complete, on all systems, place all paths to the CF online with the `VARY PATH(cfname, chpxx, cfname, chpyy . . .), ONLINE` command.

Verify the system connections to the CF with the following system commands:

```
DISPLAY XCF, CF, CFNAME=cfname
ROUTE *ALL, DISPLAY CF, CFNAME=cfname
```

2. Take the CF out of maintenance mode to permit structure allocation with the `SETXCF STOP, MAINTMODE, CFNAME=cfname` command.

Verify the permission to allocate structures in the CF can be verified with the `DISPLAY XCF, CF, CFNAME=cfname` command.

3. See "Using `REALLOCATE` or `POPULATECF` to relocate coupling facility structures" in [Chapter 4, "Managing coupling facility resources,"](#) on page 41 for a description of how to move structures to their most preferred CF(s) after CF maintenance is complete.

Removing a coupling facility from the configuration

This topic has a procedure for removing a coupling facility (CF) from the configuration when all CF structures can be removed from the CF. If not all structures can be removed from the CF (for example, the Parallel Sysplex configuration is a global resource serialization star complex with a single usable CF), then see ["Transitioning from a Parallel Sysplex to a base sysplex"](#) on page 436.

Take the following actions to delete a CF from the CFRM active policy:

1. Define a new CFRM policy that does not include any reference to that CF - this includes the CF definition and the CF name in structure preference lists. When removing the CF from a preference list, ensure the preference list contains the CF into which the structure will be rebuilt (or duplexed).

When the CF is being removed only temporarily, use a name for the new CFRM policy that is different from the name of the current CFRM policy so that the CF can be restored to the sysplex using the original/current policy.

2. CHPIDs being used for connectivity between all systems and the CF should be recorded at this point. The CHPIDs used for connectivity between all systems and the CF can be obtained using the `ROUTE *ALL, DISPLAY CF, CFNAME=cfname` command.

Information about CF connectivity to systems and CF characteristics is returned in message IXL150I. Record the following information that uniquely identifies the CF to each system:

- NODE DESCRIPTOR (type.mfg.plant.sequence)
- PARTITION
- CPCID
- CHPID 's (listed under SENDER PATH heading)

3. Activate the new policy with the `SETXCF START, POLICY, TYPE=CFRM, POLNAME=polname2` system command.

The policy changes to remove the CF definition and the policy changes to the preference list of allocated structures may become pending. Verify the deletion or pending deletion of the CF from the CFRM active policy can be verified with the `DISPLAY XCF, CF, CFNAME=cfname` system command.

4. Remove all structures from the CF. See ["Removing structures from a coupling facility for shutdown"](#) on page 437 for guidelines on removing all structures from a CF that is pending deletion from the CFRM active policy.

Unless `REALLOCATE` is used, any change to the preference list of a structure that was not allocated in the CF being removed might remain pending.

After all structures are removed from the CF, the CF will be deleted from the active policy. Deletion of the CF from the active policy can be verified with the `DISPLAY XCF,CF,CFNAME=cfname` system command.

- Using the CHPIDs from a preceding step, on all systems, place all paths to the CF offline with the `CONFIG CHP(chpxx,chpyy...),OFFLINE` system command.

Verify the unavailability of the CF on all systems with the `DISPLAY M=CHP(chpxx,chpyy...)` system command.

- SHUTDOWN the CF.

The CF can be removed from the I/O configuration when preceding actions are complete.

If making changes to CF elements (CF control units or CF channel paths) in the I/O configuration, ensure that `SOFT=VALIDATE` is specified on the `ACTIVATE` system command. `SOFT=VALIDATE` is a requirement in all N-1 partitions when changes to CF elements are made.

See [z/OS HCD Planning](#) and [z/OS HCD User's Guide](#) for additional information on deleting a CF from an I/O configuration.

See “Adding a Coupling Facility” on page 429 for a procedure to restore the CF to the sysplex.

Upgrading or replacing a coupling facility

This topic has a procedure for upgrading/replacing a coupling facility (CF) when all CF structures can be removed from the CF and the hardware identification of the CF will change. If the CF hardware identification will not change, see “Shutting down a coupling facility for maintenance” on page 431. If not all structures can be removed from the CF (for example, the Parallel Sysplex configuration is a global resource serialization star complex with a single usable coupling facility), then follow the instructions in “Upgrading or replacing all coupling facilities” on page 434.

Take the following actions before upgrading/replacing a CF:

- Define a new CFRM policy that replaces the CF hardware identification information with the information for the new CF. Use a name for the new CFRM policy that is different from the name of the active CFRM policy so that the original policy can be restored easily if a problem occurs. See “Planning a coupling facility policy” on page 49 for more information.

The same name cannot be reused unless a new CFRM couple data set is formatted and a new policy used. If the couple data set is going to be reused, then in the new CFRM policy a new name must be given to the CF that is being replaced. If a different name is used for the new CF, structure preference lists will also need to be updated.

- CHPIDs being used for connectivity between all systems and the CF should be recorded at this point. The CHPIDs used for connectivity between a system and a CF can be obtained using the `ROUTE *ALL,DISPLAY CF,CFNAME=cfname` system command.

Information about CF connectivity to systems and CF characteristics is returned in message IXL150I. Record the following information that uniquely identifies the CF to each system:

- NODE DESCRIPTOR (type.mfg.plant.sequence)
- PARTITION
- CPCID
- CHPID's (listed under SENDER PATH heading)

- Activate the new CFRM policy with the `SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname2` system command.

Assuming at least one structure is allocated in the CF, the policy changes to replace the CF definition will become pending. Replacement or pending deletion of the CF definition from the CFRM active policy can be verified with the `DISPLAY XCF,CF,CFNAME=cfname` system command.

- Remove all structures from the CF. See “Removing structures from a coupling facility for shutdown” on page 437 for guidelines on removing all structures from a CF that is in maintenance mode.

Unless REALLOCATE is used, any change to the preference list of a structure that was not allocated in the CF being removed might remain pending.

Once all structures are removed from the CF, the CF definition in the CFRM active policy will be replaced with the definition that uses the new hardware identification. Replacement of the CF definition in the active policy should be verified with the `DISPLAY XCF,CF,CFNAME=cfname` system command.

5. Using the CHPIDs from a preceding step, on all systems, place all paths to the CF offline with the `CONFIG CHP(chpxx,chpyy...),OFFLINE` system commands:

Verify the CHPIDs on all systems with the `DISPLAY M=CHP(chpxx,chpyy...)` system commands.

6. SHUTDOWN the old CF.

CF upgrade/replacement can be performed when preceding actions are complete.

The CF upgrade/replacement may involve hardware changes or changes to the I/O configuration. See *z/OS HCD Planning* and *z/OS HCD User's Guide* for more information on adding and removing coupling facilities from the I/O configuration.

If making changes to CF elements (CF control units or CF channel paths) in the I/O configuration, ensure that `SOFT=VALIDATE` is specified on the `ACTIVATE` system command. `SOFT=VALIDATE` is a requirement in all N-1 partitions when changes to CF elements are made. Changes to CF elements can be verified on all systems with the system command(s)

```
DISPLAY CF
DISPLAY M=CHP(chpxx,chpyy...)
```

Use the following procedure to bring the upgraded/replacement CF into use when upgrade/replacement is complete:

1. Ensure CF CHPIDs are online. Use the `CONFIGURE CHPID ONLINE` command on a CF to configure a CF CHPID online.

Verify CHPIDs in use with the following commands:

```
DISPLAY CHPIDS
DISPLAY RESOURCES
```

See *PR/SM Planning Guide* for more information on coupling facility control code commands.

2. Place all paths to the CF online with the `CONFIG CHP(chpxx,chpyy...),ONLINE` system command.

Verify the availability of the CF on all systems with the following system command:

```
DISPLAY XCF,CF,CFNAME=cfname
DISPLAY M=CHP(chpxx,chpyy...)
DISPLAY CF
```

3. The CF will not contain any structures. See "Using REALLOCATE or POPULATECF to relocate coupling facility structures" in Chapter 4, "Managing coupling facility resources," on page 41 for a description of how to move structures to their most preferred CFs after CF maintenance is complete.

Upgrading or replacing all coupling facilities

It is not recommended to upgrade or replace all coupling facilities in a Parallel Sysplex because some important data in a CF might be lost. IBM recommends a Parallel Sysplex configuration that allows CF maintenance or reconfiguration to be performed one CF at a time. If necessary, add at least one alternate CF to the Parallel Sysplex and following the steps in ["Upgrading or replacing a coupling facility" on page 433](#). In order to upgrade or replace all coupling facilities at the same time, a sysplex-wide IPL may be needed.

Take the following actions before upgrading/replacing all coupling facilities:

1. Define a new CFRM policy that replaces the current CF hardware identification information with the information for the new CFs. Use a name for the new CFRM policy that is different from the name of the current active CFRM policy so that the original/current policy can be restored easily if a problem occurs. See [“Planning a coupling facility policy” on page 49](#) for more information.

Note that the same names can be used for the new CFs. If a different name is used for a new CF, structure preference lists will also need to be updated.

Use the XCF couple data set format utility to create new CFRM couple data sets that can be used for sysplex re- initialization.

Use the XCF administrative data utility to define the new CFRM policy in the new CFRM couple data sets in addition to the current CFRM CDSs. Use the DSN keyword in the Administrative Data Utility control statements to define the policy in the new CFRM couple data sets. Omit the DSN keyword to define the policy in the current CFRM CDSs.

Create a new COUPLExx SYS1.PARMLIB member that specifies the new CFRM CDSs and the new CFRM policy (with the CFRMPOL keyword), which will be used for sysplex re-initialization. Other SYS1.PARMLIB members might need to be created as well:

- IEASYSxx - to point to the new COUPLExx
- LOADxx - to point to the new IEASYSxx and IODF

Note: It is recommended that there be alternate systems available to modify SYS1.PARMLIB if a modification is necessary after the sysplex is shut down.

2. Activate the new CFRM policy with the SETXCF START, POLICY, TYPE=CFRM, POLNAME=polname2 system command.

Assuming at least one structure is allocated in the CF, the policy changes to replace the CF definition and the policy changes to the preference list of allocated structures will become pending. Replacement of hardware identification or pending deletion of the named CF from the CFRM active policy can be verified with the DISPLAY XCF, CF system command.

3. Remove all structures from the CFs, at least those that might become persistent (that is, remain allocated without an active connector). See [“Deleting structures” on page 440](#) for guidelines on removing structures when allocation is not permitted in any CF.

The following structures do not need to be removed from the CFs:

- BatchPipes®
- Global Resource Serialization Lock (ISGLOCK)
- IMS Fastpath MADSIOT
- IMS OSAM and VSAM Cache
- Enhanced Catalog Sharing
- RACF Database Cache
- TCP/IP Structure for Sysplex-Wide Security Associations (SWSA)
- TCP/IP Structure for Sysplexports
- VSAM RLS Cache
- WLM (IRD and Enclaves)
- XCF Signaling

If all structures can be removed from the CF, the CFRM policy change will complete, and there is no need to continue with the steps that follow, then the only remaining step is to activate a new IODF, if necessary (ensure SOFT=VALIDATE is used on N-1 systems). The structures still allocated in the CFs can be determined by the output of the DISPLAY XCF, CF, CFNAME=ALL system command.

Any structure still allocated in the CF must be removed if it is (or might become) persistent (that is, remain allocated without an active connector). For each structure allocated in the CF, verify that no structure has a STRDISP of KEEP and that no connector to a structure

has a CONDISP of KEEP. This should be verified through the output of the DISPLAY XCF,STR,STRNAME=ALL,STAT=ALLOCATED,CONNAME=ALL system command.

4. Shut down all systems in the sysplex. Removal of all systems from the sysplex can be initiated with the ROUTE *ALL,VARY XCF,&SYSNAME,OFFLINE system command.
5. Perform the required maintenance to connect and activate the new CFs. See the *PR/SM Planning Guide* for more information.
6. Re-IPL systems with the new COUPLExx and IODF needed for the new CFs.

IXC289D might be issued to request verification to use different CFRM CDSs from what was last used. Reply to verify that the CFRM CDSs specified in the new COUPLExx SYS1.PARMLIB member should be used.

Note: The IXC289D message might have an automated reply, such as by being in the default AUTORxx parmlib member, which might not allow the operator enough time to input a reply. Therefore, carefully consider whether automating the reply to the IXC289D message as the default is appropriate for your sysplex.

Transitioning from a Parallel Sysplex to a base sysplex

This topic provides a procedure to convert from a Parallel Sysplex configuration to a base sysplex configuration (a sysplex with no coupling facilities). This procedure should only be needed when it is the intent to permanently remove all coupling facilities from the configuration. By removing use of the CFRM CDSs from the sysplex (as opposed to replacing the CFRM CDS), the use of coupling facilities will be disabled and the base sysplex will remain.

An important fact to consider is that once a CFRM CDS has been added to a sysplex, the data set cannot simply be deleted. Deleting a CFRM CDS will result in a sysplex-wide wait state, even if a CFRM policy was never activated.

Also be aware that once the CFRM couple data sets have been removed, the sysplex CDS should be reformatted for the resultant base sysplex. The reformat process will eliminate any residual information about CFRM.

To remove the CFRM CDSs from a sysplex:

1. Reconfigure any sysplex function that has a dependency on the presence of a CF. The following are some examples:
 - Take DB2 out of datasharing mode.
 - Convert from global resource serialization star complex to a global resource serialization ring complex. See [z/OS MVS Planning: Global Resource Serialization](#) for more information.
 - Convert System logger to use DASD-only logging.
 - Provide XCF signaling connectivity through CTC's
 - Stop using VTAM generic resources.

The guidelines in [“Removing structures from a coupling facility for shutdown”](#) on page 437 provides information on how to remove dependencies on the presence of a CF. Refer to the documentation for the specific application for more information.

2. Prevent any new coupling facility structure allocation with the SETXCF STOP,POLICY,TYPE=CFRM system command.
3. It is recommended to remove any remaining CF structures that may become persistent (that is, remain allocated without an active connector) at this point in order to minimize loss of coupling facility data. See [“Deleting structures”](#) on page 440 for guidelines on removing structures when structure allocation is not permitted.

The following structures do not need to be removed from the CFs:

- BatchPipes
- Global Resource Serialization Lock (ISGLOCK)

- IMS Fastpath MADSIOT
 - IMS OSAM and VSAM Cache
 - Enhanced Catalog Sharing
 - RACF Database Cache
 - TCP/IP Structure for Sysplex-Wide Security Associations (SWSA)
 - TCP/IP Structure for Sysplexports
 - VSAM RLS Cache
 - WLM (IRD and Enclaves)
 - XCF Signaling
4. Create a COUPLExx SYS1.PARMLIB member that does not specify a CFRM CDS (or a CFRMPOL) and does not use XCF signaling structures (use only CTC devices for XCF signaling connectivity). Other SYS1.PARMLIB members might need to be created as well:
- IEASYSxx - to point to the new COUPLExx and to use GRS=TRYJOIN
 - LOADxx - to point to the new IEASYSxx
- Note:** It is recommended that there be alternate systems available to modify the affected system's parmlib if a modification is necessary after the affected system is shut down.
5. Shut down all images and re-IPL the systems with newly formatted sysplex couple data sets using the new SYS1.PARMLIB members. This can be accomplished in either of the following ways:
- Shut down all images in the sysplex and reformat the current sysplex CDS(s) from a system that has DASD access to those data sets.
 - Format new sysplex couple data sets using IXCL1DSU and modify the COUPLExx sysplex CDS statements to point to the newly formatted couple data sets. Delete the old sysplex couple data sets after all systems in the sysplex are re-IPLed and are therefore no longer using the old sysplex couple data sets.
- Do **not** use the SETXCF command to switch to the new sysplex couple data sets.
6. The old CFRM CDSs can be deleted when XCF indicates that either CFRM is not defined to the sysplex or CFRM is not in use by any system. To determine if the CFRM CDSs can be deleted, use the DISPLAY XCF, COUPLE, TYPE=CFRM system command.

Removing structures from a coupling facility for shutdown

This topic assumes that steps have been taken so that allocation is not permitted in the coupling facility (for example, maintenance mode or delete pending). Verify that allocation is not permitted in the CF with the DISPLAY XCF, CF, CFNAME=cfname system command.

For structures that support rebuild, a rebuild process can be used to remove a structure from a CF. Structures can be removed individually through operations for an individual structure, as a group through operations that can target all structures in a CF, or through a REALLOCATE process. See "Using REALLOCATE or POPULATECF to relocate coupling facility structures" for more information on initiating a REALLOCATE process to remove all structures from a CF that does not permit allocation. A REALLOCATE process can be started with the SETXCF START, REALLOCATE system command.

An alternative to a REALLOCATE process is to use the following commands to remove all structures from a CF:

- Rebuild simplex structures into an alternate CF with the SETXCF START, REBUILD, CFNAME=cfname system command.
- Remove duplexed structure instances from the CF with the SETXCF STOP, REBUILD, DUPLEX, CFNAME=cfname system command.
- Rebuild each XCF signaling structure with the SETXCF START, REBUILD, STRNAME=stiname, LOCATION=OTHER system command.

To process each structure individually, see [“Actions to remove a structure from a coupling facility” on page 438](#) for each structure.

For structures for which rebuild is not possible, see [“Actions to remove a structure from a coupling facility” on page 438](#). If a structure is pending deletion or in transition, restore connectivity to the CF. If connectivity cannot be restored to the CF, the structure can only be removed by deleting the CF from the CFRM active policy.

Verify the removal of all structures from the CF with the `DISPLAY XCF,CF,CFNAME=cfname` system command.

Removing a structure from a coupling facility

To move a CF structure instance from one CF to another CF:

1. Ensure the structure has another CF in its preference list - the structure may only be allocated in coupling facilities that are in its preference list. If necessary, create a new CFRM policy (or update an existing CFRM policy) to add the target CF to the preference list for the structure that is to be moved. See [“Planning a coupling facility policy” on page 49](#) for more information.

It might also be necessary or desirable to remove the CF in which the structure is currently allocated from the structure's preference list, particularly for the case where the structure does not support rebuild. Note that it is generally recommended that a structure always have at least two coupling facilities in its preference list.

Activate the new/updated CFRM policy using the `SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname` system command.

The policy change of the structure preference list might become pending until the structure is moved to another CF in the next step.

2. Use the instructions in [“Actions to remove a structure from a coupling facility” on page 438](#) to move the structure instance from one CF to another.

Verify an update to the structure preference list in the CFRM active policy with the `DISPLAY XCF,STRUCTURE,STRNAME=strname` system command.

If the structure is no longer in use (that is, the structure was deleted), use of the structure can be re-initiated once the preference list in the CFRM active policy is updated to remove the CF.

Verify the CF in which the structure is allocated with the `DISPLAY XCF,STRUCTURE,STRNAME=strname` system command.

Actions to remove a structure from a coupling facility

There are several ways an active structure instance may be removed from a CF:

- Stop structure duplexing - keeping only the structure instance in the other CF
- Rebuild the structure into another CF
- Delete the structure entirely (use only as the last resort)

The method to remove a structure instance from a CF depends on the state of the structure. The state of the structure is provided in the output of the `DISPLAY XCF,STRUCTURE,STRNAME=strname` system command.

A duplexed CF structure has two structure instances allocated, in two different coupling facilities. To delete one of the structure instances from a particular CF, stop duplexing and keep the structure instance in the other CF.

- When the duplexed structure instance to be deleted is listed as the `DUPLEXING REBUILD OLD STRUCTURE`, stop duplexing to keep the new structure with the `SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=NEW` system command.

- When the duplexed structure instance to be deleted is listed as the DUPLEXING REBUILD NEW STRUCTURE, stop duplexing to keep the old structure with the SETXCF STOP, REBUILD, DUPLEX, STRNAME=stname, KEEP=OLD system command.

Note that if duplexing is stopped for a user-managed duplexed structure that does not have an active connector (for example, a DB2 cache structure for Group Buffer Pools), the structure cannot be re-duplexed (until a connection becomes active).

An active simplex CF structure may be able to be rebuilt into another CF. See instructions as in [“Rebuilding a structure into another coupling facility”](#) on page 439.

When an active simplex structure cannot be rebuilt into another CF, the only other alternative is to delete the structure. See [“Deleting structures”](#) on page 440.

If a structure dump exists for the structure, consider making a dump data set available to capture the dump. If an SVC dump is not in progress, deletion of the structure dump can be forced in order to allow the structure to be deallocated with the SETXCF FORCE, STRDUMP, STRNAME=stname, STRDUMPID=stidumpid system command.

If a structure is pending deletion or in transition, restore connectivity to the CF. If connectivity cannot be restored to the CF, the structure can only be removed by deleting the CF from the CFRM active policy. See [“Removing a coupling facility from the configuration”](#) on page 432.

Rebuilding a structure into another coupling facility

All CF structures for IBM software support a rebuild process (either user-managed or system-managed). Before rebuilding a structure into an alternate CF, ensure the alternate CF has sufficient connectivity and memory capacity. The alternate CF must be in the preference list of the structure to be rebuilt. If necessary, create a new CFRM policy (or modify an existing CFRM policy) with the alternate CF in the structure preference list and activate it. See [“Planning a coupling facility policy”](#) on page 49 for more information. Start a CF structure rebuild process that will rebuild the structure into another CF using the SETXCF START, REBUILD, STRNAME=stname, LOCATION=OTHER system command.

Rebuilding a structure might take several minutes to complete. A response to the rebuild request indicates if the rebuild has been successfully initiated, and a message will be issued when the initiated rebuild processing actually completes.

To perform this function for multiple structures, REALLOCATE and rebuild by CFNAME can also be used. See also [“Using REALLOCATE or POPULATECF to relocate coupling facility structures”](#) in [Chapter 4](#), [“Managing coupling facility resources,”](#) on page 41.

Additional processing for rebuilding structures

Structures that can be rebuilt might require additional processing to move them to an alternate CF. Rebuild scenarios that require additional processing are:

- The structure has no active connections; a structure can only be rebuilt without any active connections via the system-managed rebuild process, which in turn requires that the primary CFRM CDS in use is formatted with system-managed rebuild process support (SMREBLD).
- The structure has failed-persistent connections. The existence of failed-persistent connections might prevent a user-managed rebuild from being processed successfully by the connectors.

Allocated structures with no connections are listed in the output of the DISPLAY XCF, STRUCTURE, STATUS=(NOCONN) system command.

Structures with a failed-persistent connection are listed in the output of the DISPLAY XCF, STRUCTURE, STATUS=(FPCONN) system command.

System-managed rebuild is required for structures with no active connections (no connections or only failed-persistent connections). Ensure that system-managed rebuild is supported by the primary CFRM CDS, or initialize the application associated with the structure to create an active connector to the structure.

For structures with a failed-persistent connection, recovery actions should be handled by the owning application. Perform any required actions (for example, initialize the application instances) associated with each failed-persistent connection to initiate recovery actions. See applications specific instructions for more information on required actions to resolve failed-persistent connections.

For information on handling rebuild problems, see [“Structure rebuild recovery” on page 448](#).

Deleting a structure from the CFRM active policy

Before removing an allocated structure, read the instructions in [“Actions to remove a structure from a coupling facility” on page 438](#) in order to better understand the implications.

To delete a structure from the CFRM active policy:

1. Create a new CFRM policy (or modify an existing CFRM policy) to delete the structure definition.
2. Activate the new or updated CFRM policy using the SETXCF
START,POLICY,TYPE=CFRM,POLNAME=polname system command.

The policy change to delete the structure definition will become pending if the structure is allocated. Verify the deletion or pending deletion of the structure from the active policy with the DISPLAY
XCF,STRUCTURE,STRNAME=strname system command.

3. Follow the instructions below on [“Deleting structures” on page 440](#) for the recommended procedures for discontinuing use of the structure that is to be removed. The deletion of the structure from the active policy can be verified with the DISPLAY XCF,STRUCTURE,STRNAME=strname system command.

Deleting structures

Deleting an active structure for CF reconfiguration or maintenance should only be considered when the structure cannot be relocated to an alternate CF. For example, although IBM does not recommend relying on a single CF in the Parallel Sysplex, a configuration with a single CF provides no alternate CF into which structures can be rebuilt.

The procedures in the following sections describe how to delete active IBM software structures. For structures not listed, similar information should be provided in the application documentation. Verify the deletion of the active structure instances with the DISPLAY XCF,STRUCTURE,STRNAME=strname system command.

If a structure dump exists for the structure, consider making a dump data set available to capture the dump. If an SVC dump is not in progress, deletion of the structure dump can be forced in order to allow the structure to be deallocated with the SETXCF
FORCE,STRDUMP,STRNAME=strname,STRDUMPID=strdumpid system command.

If a structure is pending deletion or in transition, restore connectivity to the CF. If connectivity cannot be restored to the CF, the structure can only be removed by deleting the CF from the CFRM active policy. See [“Removing a coupling facility from the configuration” on page 432](#).

Deleting the JES2 checkpoint structure

1. Use the JES2 reconfiguration dialog to move the checkpoint data set to another CF or to DASD. To use the dialog, ensure that you have defined a NEWCKPTx in the JES2 CKPTDEF initialization statement in one of the following ways:
 - JES2 initialization parameters
 - Dynamically through the \$T CKPTDEF command
2. Issue the following JES2 command to switch to the alternate checkpoint data set:

```
$T CKPTDEF,RECONFIG=YES
```

Follow the reconfiguration dialog prompts.

When finished, issue the following JES2 display command to verify that the structure is not still in use by JES2:

```
$D CKPTDEF
```

3. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the system command:

```
SETXCF FORCE,STRUCTURE,STRNAME=stname
```

4. If necessary, change the JES2 initialization parameters to point to the new checkpoint data set for the next JES2 warm start.

To re-initiate use of the JES2 checkpoint structure, issue the JES2 command:

```
$T CKPTDEF,RECONFIG=YES
```

Follow the reconfiguration dialog prompts.

Deleting DB2 cache structures for group buffer pools

1. Stop all DB2 members in the data sharing group.

If it is not possible to stop all the DB2 members, see *DB2 Data Sharing: Planning and Administration* for information. The following command can be used for a normal shutdown of all DB2 members:

```
STOP DB2 MODE(QUIESCE)
```

DB2 list (SCA) structures are also deleted with the preceding command.

2. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the system command:
SETXCF FORCE,STRUCTURE,STRNAME=stname

Restart DB2 to re-initiate use of the structure.

Deleting the DB2 list (SCA) structure

1. Use the IRLM status command to determine if any DB2 member that is in the data-sharing group has retained locks. If a member has retained locks, reinitialize the associated DB2 member to recover the retained locks.
2. Initiate a normal shutdown of all DB2 members with the following command:

```
STOP DB2 MODE(QUIESCE)
```

The DB2 cache structures for the group buffer pools are also deleted automatically.

3. Deletion of the structure may need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE,STRUCTURE,STRNAME=stname system command.

For detailed information, see *DB2 Administration Guide* and *DB2 Data Sharing: Planning and Administration*.

Deleting CICS temporary storage queue pool structures

1. To delete a CICS structure for a temporary storage (TS) queue pool, stop all TS queue pool servers for the structure. If there are no connections to the structure, use the STOP system command to stop the TS server. If there are connections, disconnect active CICS connectors using the MODIFY server,CANCEL command (where server is the name of the TS queue pool server).
2. When all servers are disconnected, if the existing data needs to be saved, unload the structure contents to a sequential data set by running the queue pool server program DFHXQMN with the FUNCTION=UNLOAD option as described in the *CICS System Definition Guide*.

3. Deletion of the structure may need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=DFHXQLS_poolname system command.

The saved structure contents can be reloaded later from the sequential data set into a new structure by running the queue pool server program DFHXQMN with the FUNCTION=RELOAD option.

Deleting CICS coupling facility data table structures

1. To delete a CICS structure for a coupling facility data table (CFDT) pool, stop all CFDT pool servers for the structure. If there are no connectors to the structure, use the STOP system command to stop the CFDT server. If there are connectors, issue the MODIFY server, CANCEL system command to disconnect active CICS connectors (where server is the name of the CFDT pool server).
2. When all servers are disconnected, if the existing data needs to be saved, unload the structure contents to a sequential data set by running the CFDT pool server program DFHCFMN with the FUNCTION=UNLOAD option as described in the *CICS System Definition Guide*.
3. Deletion of the structure may need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=DFHCFLS_poolname system command.

The saved structure contents can be reloaded later from the sequential data set into a new structure by running the queue pool server program DFHCFMN with the FUNCTION=RELOAD option.

Deleting CICS named counter structures

1. To delete a CICS structure for a named counter (NC) pool, stop all named counter pool servers for the structure. If there are no connections to the structure, use the STOP system command to stop the named counter server. If there are connections, issue the MODIFY system command to disconnect active CICS connections: F server, CANCEL where server is the name of the named counter pool server.
2. When all servers are disconnected, if the existing data needs to be saved, unload the structure contents to a sequential data set by running the named counter pool server program DFHNCMN with the FUNCTION=UNLOAD option as described in the *CICS System Definition Guide*.
3. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=DFHNCLS_poolname system command.

The saved structure contents can be reloaded later from the sequential data set into a new structure by running the named counter pool server program DFHNCMN with the FUNCTION=RELOAD option.

Deleting the RACF database cache structures

Delete the RACF database cache structure by removing RACF from data sharing mode. Remove RACF from data sharing mode with the RVARY NODATASHARE RACF command.

If the installation has defined RACF as a subsystem in the IEFSSNxx SYS1.PARMLIB member, the command can be issued from a z/OS operator console, but the subsystem recognition character that the installation has defined through the command prefix facility will need to be used. Otherwise, the command must be issued from an authorized TSO/E user ID. For more information on IEFSSNxx, see [z/OS MVS Initialization and Tuning Reference](#).

Use of the structure can be re-initiated when RACF is restored to data sharing mode with the RVARY DATASHARE RACF command.

Deleting IMS CQS log stream structures

- Shutdown normally any IMSs connected to the CQSs using the structure.
- If any CQS address spaces are active after IMS shutdown, stop the CQS address spaces with the STOP cqsjobname system command.
- The log stream structures associated with the CQS structures should be deleted when the last CQS disconnects from the z/OS system logger.

Deleting IMS CQS message queue (EMHQ, EMHQ overflow, MSGQ, and MSGQ overflow) structures

1. Shutdown normally any IMSs connected to the CQSs using the structure.

If any CQS address spaces are active after IMS shutdown, stop the CQS address spaces with the STOP cqsjobname system command.

2. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Ensure that the strname in this command is the same as the strname specified in the CQS global structure definition PROCLIB member and the CQS local structure definition PROCLIB member.

Deleting the IMS fastpath MADSIOT structure

Shutdown any IMS subsystems using the structure.

Deleting IMS OSAM and VSAM cache structures

Shutdown each IMS subsystem using the structure.

Deleting IMS CQS resource structures

1. Shutdown normally any IMSs and RMs connected to the CQSs using the structure.

If any CQS address spaces are active after IMS shutdown, stop the CQS address spaces with the STOP cqsjobname system command.

2. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Ensure that the strname in this command is the same as the strname specified in the CQS global structure definition PROCLIB member and the CQS local structure definition PROCLIB member.

Deleting the IMS fastpath VSO cache structure

Shutdown any IMS's using the structure.

Deletion of the structure may need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Deleting the IRLM lock structure

1. If an IMS or DB2 subsystem retains a lock, restart that IMS or DB2 subsystem to clean up the locks.

It can be determined if any IMS or DB2 subsystem in the data sharing group retains a lock with the MODIFY ir1mproc, STATUS, ALLD system command (where ir1mproc is the IRLM associated with IMS or DB2).

2. Initiate a normal shutdown of all IMS or DB2 subsystems identified with IRLM to cause IRLM to disconnect from the data sharing group.
3. For DB2, the IRLM is normally auto-started and will stop automatically when DB2 terminates. For IMS and for DB2 IRLMs that do not auto-stop, issue the STOP ir1mproc system command.

Make sure that it terminates successfully.

4. Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE” on page 447](#) for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Deleting the XCF signaling structure

1. Ensure that redundant XCF signaling paths (CTCs or other CF structures) are started and ensure that they have capacity to ensure that system communications in the Parallel Sysplex are not disrupted when the structure is deleted.
2. Stop XCF signaling through the structure with the following commands:

```
ROUTE *ALL,SETXCF STOP,PATHOUT,STRNAME=stname
ROUTE *ALL,SETXCF STOP,PATHIN,STRNAME=stname
```

Stop PATHOUT before PATHIN to avoid delays due to additional outbound signal activity on the structure that would need to be resolved prior to the structure being deleted.

To re-initiate use of the structure, issue the following commands on all systems as applicable:

```
SETXCF START,PATHIN,STRNAME=stname
SETXCF START,PATHOUT,STRNAME=stname
```

Deleting the system logger structures

1. If any subsystem or application is using the system logger, provide a normal shutdown of the subsystem and application. Refer to the topic "Finding Information for System Logger Applications" in Chapter 10, "Planning for system logger applications," on page 213 for more specific information on logger exploiter use.

Verify the users of system logger structures with the following system commands:

```
DISPLAY LOGGER,STR
DISPLAY XCF,STRUCTURE,STRNAME=stname
DISPLAY LOGGER,L,LSN=streamname
```

Action will need to be taken on each system that has log stream connections and is using the CF structure.

2. For OPERLOG, use the D C or D C,HC commands to determine if a log stream is being used for Consoles. If so, enter the VARY OPERLOG,HARDCPY,OFF command to cause consoles to disconnect from its log stream and change the hardcopy medium.

Also, exit from all SDSF sessions that are browsing from OPERLOG.

After the structure has been deleted, the VARY OPERLOG,HARDCPY command can be used to cause consoles to use a log stream again.

3. For LOGREC, use the D LOGREC command to determine if a log stream is being used. If so, enter the SETLOGRC DATASET command (assuming a data set was established when the system was IPLed) to cause LOGREC to disconnect from its log stream and stop LOGSTREAM recording of LOGREC entries.

After the structure has been deleted, the SETLOGRC LOGSTREAM command can be used to cause LOGREC to use a log stream again.

4. After the Logger users have disconnected from their log streams, you can use D LOGGER,C,LSN=streamname commands to determine if logger has completed its logstream disconnect processing for the log streams.

If the log stream disconnect status remains in DISCONNECT PENDING state, then the SETLOGR FORCE,DISConnect,LSName=streamname command can aid in getting the log stream disconnected from the system and for logger to disconnect from the CF structure.

5. However, after logger has disconnected from the CF structures, check if any are still in a failed-persistent connection state using the XCF,STRUCTURE,STRNAME=structname command.

Deletion of the structure might need to be forced. See "Deleting structures with SETXCF FORCE" on page 447 for more information on forcing deletion of persistent structures with the SETXCF FORCE,STRUCTURE,STRNAME=stname system command.

Note that forcing the structure connections and the subsequent deletion of a structure used by logger can result in logger marking the log stream as being *damaged*, meaning there may be a *possible loss*

of data condition for the log stream. Check on how each log stream exploiter deals with a *damaged* log stream condition to ensure forcing the structure deletion is appropriate.

6. After all users of system logger have disconnected from their log streams and logger has disconnected from a CF structure, the system deletes the structure in the CF.

Deleting the VSAM RLS lock and cache structures

1. Before shutting down VSAM RLS processing, follow your installation procedures for shutting down applications that use open VSAM data sets in RLS mode. For CICS, use one of the following ways to quiesce the data sets:

- Use CEMT to shut down all CICS subsystems
- Issue EXEC CICS, SET DSNAME(name) QUIESCED

Some CICS subsystems can be shut down by issuing CEMT P SHUT.

2. Issue the following command to shut down VSAM RLS processing on all systems: V SMS, SMSVSAM, TERMINATESERVER
3. Wait for all VSAM RLS address spaces (SMSVSAM) to end.
4. Deletion of the structure IGWLOCK00 might need to be forced. See [“Deleting structures with SETXCF FORCE”](#) on page 447 for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=IGWLOCK00 system command.

Alternatively, use the VARY SMS, SMSVSAM, FORCEDELETELOCKSTRUCTURE system command.

Use of the structure can be re-initiated with the VARY SMS, SMSVSAM, ACTIVE system command.

VSAM RLS processing is again available. To make available for record-level sharing any data set with unresolved locks (lost-locks condition), perform all necessary forward and backward recovery for the data sets.

Note: When a lost-locks condition exists for CICS, there is no need to perform forward recovery. CICS automatically performs backout recovery when it is restarted.

Deleting the global resource serialization lock structure

To delete the global resource serialization lock structure (ISGLOCK), a sysplex-wide IPL is needed to convert to a global resource serialization ring complex. After all systems are reset, re-IPL them with GRS=TRYJOIN. See [z/OS MVS Planning: Global Resource Serialization](#) for more information.

Note that when performing a sysplex-wide IPL for CF maintenance or reconfiguration, ISGLOCK does not need to be deleted, but the CFRM active policy used at IPL must specify a usable CF for ISGLOCK in order to participate in a global resource serialization star complex.

Deleting the VTAM structure for generic resources

Stop all the VTAMs that are connected to the Generic Resource structure.

Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE”](#) on page 447 for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Deleting the VTAM structure for MNPS

Stop all the VTAMs that are connected to the Generic Resource structures.

Deletion of the structure might need to be forced. See [“Deleting structures with SETXCF FORCE”](#) on page 447 for more information on forcing deletion of persistent structures with the SETXCF FORCE, STRUCTURE, STRNAME=stname system command.

Deleting the TCP/IP structure for Sysplexports

Stop each TCP/IP stack that is connected to the Sysplexports structure. When all stacks have been stopped, all connections to the structure will have been disconnected, and the structure will be deleted.

Deleting the TCP/IP structure for sysplex-wide security associations (SWSA)

Stop each TCP/IP stack that is connected to the SWSA structure. When all stacks have been stopped, all connections to the structure will have been disconnected, and the structure will be deleted.

Deleting the enhanced catalog sharing structure

Initiate normal connector disconnect with the `MODIFY CATALOG, ECSHR (DISCONNECT)` command. See [z/OS DFSMS Managing Catalogs](#) for more information. Use of the structure can be re-initiated with the `MODIFY CATALOG, ECSHR (AUTOADD)` system command.

Deleting the DFSMSHsm common recall queue structure

The following information can be found under the topic "Correcting Errors within the Common Recall Queue" of the [z/OS DFSMSHsm Storage Administration](#).

1. Move all recall requests from the CRQ back to the local DFSMSHsm hosts for processing by issuing the following command on all DFSMSHsm hosts that are connected to the CRQ structure:

```
SETSYS COMMONQUEUE(RECALL(DISC))
```

Each host will issue message ARC1502I after it disconnects from the structure.

To determine if all DFSMSHsm hosts have disconnected, issue the following command:

```
DISPLAY XCF,STRUCTURE,STRNAME=SYSARC_basename_RCL
```

where `basename` is the name specified on the following command:

```
SETSYS COMMONQUEUE(RECALL(CONNECT(basename)))
```

The number of connections reported should be zero. Hosts will not disconnect from the structure until they have completed processing requests that they have already selected.

2. Deletion of the structure might need to be forced. See ["Deleting structures with SETXCF FORCE"](#) on [page 447](#) for more information on forcing deletion of persistent structures with the `SETXCF FORCE, STRUCTURE, STRNAME=SYSARC_basename_RCL` system command.

To re-initiate use of the DFSMSHsm common recall queue structure, issue the following command on each DFSMSHsm host that was previously connected:

```
SETSYS COMMONQUEUE(RECALL(CONNECT(basename)))
```

This command will re-allocate the structure and cause each host to automatically move all of its local recall requests back onto the CRQ.

Deleting the WLM structures (IRD and enclaves)

There is no command to get WLM to discontinue use of the structures. However, WLM will disconnect if there is a failure accessing the structure.

Note that when performing a sysplex-wide IPL for CF maintenance or reconfiguration, the WLM structures (IRD and Enclave) do not need to be deleted.

Deleting the WMQ shared queued structure

1. Use the following command to get a list of all the queues using the CF structures to be deleted:

```
DISPLAY QUEUE(*) QSGDISP(SHARED) CFSTRUCT(structure-name)
```

2. Delete all the queues that use the structure.
3. Stop and restart each queue manager in the queue-sharing group in turn to disconnect WebSphere MQ and DB2 from the structure and delete information about it. All the queue managers do not need to be stopped at once; just one at a time.

See IBM MQ in IBM Documentation (www.ibm.com/docs/en/ibm-mq) for more information.

Deleting structures with SETXCF FORCE

The SETXCF FORCE command allows you to deallocate persistent structures (active structures with no active connections). However, use this command with caution. The SETXCF FORCE command might cause a loss of structure data, so be sure you understand how the application is using the structure. See [“A Review of Some Basic Concepts” on page 425](#) for more information. The following command will cause an active structure with no active connections to be deallocated, and any failed-persistent connections will be deleted:

```
SETXCF FORCE,STRUCTURE,STRNAME=stname
```

For information about the FORCE option on MVS commands, see [z/OS MVS System Commands](#).

Dynamic IO Coupling Facility Changes

Modifying Channel Path Access to a Coupling Facility Channel Path of Coupling channel path types

Scenario: Removing ALL images from the Lowest Level CSS from the Access List of a CF Channel Path.

When two Coupling CHPIDs are connected in an IODF, HCD establishes the connected path (CPATH) mapping of the Logical Connection between two Channel Subsystems in the definitions of both the source and the target coupling channel paths. If one of the connected coupling paths or both connected paths are spanned, the corresponding CPATH mapping points to the Lowest Level CSS ID that may have access to the peer channel path.

If the Channel Path Access and/or Candidate List of a Coupling Facility Channel path changes in such a way that ALL of the Lowest Level CSS images are removed from that Channel Path through HCD, a subsequent Dynamic Activate will result in the Channel Path on the Connected Side to be DELETED and then RE-ADDED. For additional information see [Modifying Channel Path Access to a spanned Coupling Facility Channel Path in z/OS Hardware Configuration Definition Planning](#).

Follow the best practices for a change in this nature:

1. Issue an ACTIVATE TEST with VALIDATE HW Changes on ALL images involved.
2. If the CHANNEL PATH is Connected to a CF ONLY ensure that the CHPID is CONFIGURED OFFLINE from the CF LPAR.
3. If access to an ENTIRE CF will be lost (all chpids under the control unit toward this CF were involved in the change) ensure that the CF is placed in MAINTMODE and Structures are REALLOCATED to another available CF.
 - a. SETXCF START,MAINTMODE,CFNAME=xxxx -> Place xxxx into MAINTMODE.
 - b. SETXCF START,REALLOC => Move all str currently on xxxx to yyyy.
 - c. D XCF,STR,STRNAME=ALL,STATUS=ALLOCATED to confirm no str left on xxxx any more.
 - d. Configure OFFLINE the CHPIDS that will be impacted by the CHANGE.
 - e. Activate Changes that will impact CF xxxx.
 - f. Configure ONLINE CHPIDS impacted by the Change.
 - g. SETXCF STOP,MAINTMODE,CFNAME=xxxx => Return xxxx back for usage.
 - h. SETXCF START,REALLOC => D XCF,STR,STRNAME=ALL,STATUS=ALLOCATED to confirm str should be on xxxx are there as expected
4. Make changes such that ALL Channels to a CF are not impacted within a single IODF change, so the CF can continue to be utilized during the IODF Activation. If there are four Channels Connected from an IMAGE to a CF, change two Channels at a time using two different IODF HW activates. Those two affected CHPIDs should be configured OFFLINE prior to the IODF activation. CONFIGURE these CHPIDs back ONLINE after the IODF activation.

Structure rebuild recovery

The following topic describes a recovery scenario for a structure rebuild that does not complete (a structure rebuild hang).

A structure rebuild might not complete for many reasons. The application or connector is the most likely source of the rebuild hang; however, the hang might be the result of a problem with the XES or XCF z/OS components. For information about sysplex recovery, see [z/OS MVS Diagnosis: Reference](#).

Steps for handling a structure rebuild hang

The following are general steps to follow when a rebuild hang occurs:

1. Look for and respond to any messages that indicate problems with rebuild. For example, IXL040E may be issued to identify a response that a connector has not provided.
2. To determine the state of the rebuild process, issue the following command: `DISPLAY XCF, STRUCTURE, STRNAME=strname`

The rebuild hang might be the result of a problem with one or more connections associated with the structure. For connections that have not completed the rebuild process, check the status of the system, subsystem, or address space where the connector resides and try to correct the problem. See [“Identify the connector or systems that require recovery actions” on page 453](#) for more information.

3. Collect documentation as follows:
 - Take an SDUMP of the connections that are hanging, the XCF address space, and all data spaces owned by each address space to be dumped. For additional SDATA parameters, information about the couple data set dump utility, IPCS and trace parameters. See [z/OS MVS Diagnosis: Reference](#).
 - Dump the CFRM and sysplex couple data sets.
4. Try to recover from the failure.

Stop the rebuild with the `SETXCF STOP, REBUILD, STRNAME=strname` system command. Or stop the duplexing rebuild with the `SETXCF STOP, REBUILD, DUPLEX, STRNAME=strname, KEEP=OLD` system command. Determine if the rebuild stopped using the `DISPLAY XCF, STRUCTURE, STRNAME=strname` system command. If the structure is still in a rebuild process, collect documentation as indicated in step 3.
5. If the rebuild process was stopped, attempt to rebuild the structure again.
6. If the structure has failed-persistent connections, you might need to clean up the connections before you can rebuild the structure.
7. If a connector problem exists and cannot be resolved, consider terminating the connector that is not providing a response. Terminating a connector might involve shutting down the subsystem associated with the connector, cancelling the connector address space, or varying the connector system out of the Parallel Sysplex.
8. When you have resolved the problem with the connector, attempt to rebuild the structure again.
9. With the information collected in step 3, diagnose and correct the cause of the hang.

Example of a rebuild hang

Consider a sysplex with three systems (SYSTEM1, SYSTEM2, SYSTEM3). The `SETXCF START, REBUILD` command is issued for the XCF structure IXC1, but the rebuild has not completed. The `DISPLAY XCF, STRUCTURE, STRNAME=IXC1` command is entered and the response to the command indicates that a connector to IXC1 on SYSTEM2 has not completed the rebuild process.

Figure 83 on page 449 shows the output to the commands. Information about the connector that has not responded to the rebuild request is indicated by an asterisk next to the connection name (SIGPATH_02000002 in this example). Based on the information, the system programmer is able to collect the appropriate documentation and determine the recovery action.

```

===> SETXCF START,REBUILD,STRNAME=IXC1

IXC367I THE SETXCF START REBUILD REQUEST FOR STRUCTURE 924
IXC1 WAS ACCEPTED.
IXC467I REBUILDING PATH STRUCTURE IXC1 925
      RSN: PARTICIPANT
      DIAG073: 08840001 0000000A 00000000 04000000 00000000

      /
      /
      /
      /

===> D XCF,STR,STRNAME=IXC1

IXC360I 09.53.28 DISPLAY XCF 942
STRNAME: IXC1
===> STATUS: REASON SPECIFIED WITH REBUILD START:
      OPERATOR INITIATED
      REBUILD PHASE: QUIESCE
      POLICY SIZE : 10000 K
      POLICY INITSIZE: N/A
      REBUILD PERCENT: N/A
      PREFERENCE LIST: TESTCF
      EXCLUSION LIST IS EMPTY

ACTIVE STRUCTURE
-----
ALLOCATION TIME: 08/30/95 09:50:49
CFNAME : TESTCF
CF: CF01...
      PARTITION: 0 CPCID: 00
ACTUAL SIZE : 10240 K
STORAGE INCREMENT SIZE: 256 K
VERSION : AB991335 42205400
DISPOSITION : DELETE
ACCESS TIME : 0
MAX CONNECTIONS: 32
# CONNECTIONS : 3
NEXT USYNC EVENT : 00000001
NEXT USYNC STATE : 00000000 00000000 00000000 00000000
                  00000000 00000000 00000000 00000000

* ASTERISK DENOTES CONNECTOR WITH OUTSTANDING REBUILD RESPONSE
CONNECTION NAME ID VERSION SYSNAME JOBNAME ASID STATE
-----
SIGPATH_01000001 01 00010001 SYSTEM1 XCFAS 0006 ACTIVE
===> *SIGPATH_02000002 02 00020001 SYSTEM2 XCFAS 0006 ACTIVE
      SIGPATH_03000003 03 00030001 SYSTEM3 XCFAS 0006 ACTIVE

```

Figure 83. Message Output from a Rebuild Hang

Guidelines for unplanned outages

This topic provides guidelines for recovering from CF failures. Two scenarios describe situations when another CF is available and when another CF is not available.

For additional information, you should be familiar with the following references:

- [IBM z/OS Parallel Sysplex \(www.ibm.com/products/zos/parallel-sysplex\)](http://www.ibm.com/products/zos/parallel-sysplex)
- *z/OS Parallel Sysplex Test Report*/*z/OS Parallel Sysplex Overview*

Guidelines for recovering from coupling facility failures

The following information assumes that a complete failure of a CF occurred (for example, a power failure, system reset, power-on reset, LPAR reset or deactivate). For information about how to handle structure failure and connectivity failure scenarios, see *z/OS Parallel Sysplex Test Report*. Note that CF failures are observed by the MVS systems as a loss of CF connectivity. There is no indication that distinguishes a true sysplex loss of connectivity (in which the CF stays up but the link(s) to the CF fail) from other types of CF failure, such as a power failure or system reset.

When a CF without power or battery backup fails, all of the structures and the data contained in the structures are lost. However, because these failures all appear to z/OS as loss of connectivity to the CF (in

which case the structures and data are not lost), z/OS makes the conservative assumption that the “loss of connectivity” is in fact just a loss of connectivity to the CF and thus, that the structures have not been lost.

In these CF failure scenarios, duplexed structure will failover to the structure instance in another CF. For structures that are not being duplexed, it is expected that applications that own structures on the failed CF will recover either by rebuilding those structure into an alternate CF or through some other means. There are scenarios where application recovery of structures might not be successful (for example, your sysplex might not have an alternate CF in which to rebuild structures, or a concurrent application failure prevents a successful structure rebuild).

The CASE 1 scenario describes procedures to follow when an alternate CF is available to recover from the failed CF. The CASE 2 scenario describes the situation in which an alternate CF is not available in the parallel sysplex.

CASE 1 - An alternate coupling facility is available

When an alternate CF is available, the structure might be able to be rebuilt from the failed CF to an alternate CF. The possible results of the rebuild consist of the following:

- All structures successfully rebuild into the alternate CF. See [“All structures rebuild” on page 450](#).
- Some structures are not rebuilt into the alternate CF. See [“Some structures do not rebuild” on page 450](#).
- Structures in the failed CF become persistent or structure connections become failed-persistent. See [“Structure or connection persistence occurs” on page 450](#).
- Structure rebuild fails to complete. See [“Structure rebuild fails” on page 452](#).
- Structures cannot be rebuilt because there are storage constraints in the alternate CF. See [“Coupling facility storage constraints exist” on page 452](#).

All structures rebuild

All structures successfully rebuild into the alternate CF. See "Using REALLOCATE or POPULATECF to relocate coupling facility structures" in Chapter 4, [“Managing coupling facility resources,” on page 41](#) for a description of how to move structures to their most preferred CFs after a CF has been restored to the configuration.

Some structures do not rebuild

Some applications that use CF structures do not support rebuild. Structures such as the DB2 cache structures (group buffer pools) might require manual intervention to reallocate the structures in the alternate CF. Follow application-specific recovery procedures for applications that do not support rebuild. Likewise, when the failed CF is restored, the manual procedure will be necessary to move these structures back into the restored CF - see "Using REALLOCATE or POPULATECF to relocate coupling facility structures". For additional information see *z/OS Parallel Sysplex Test Report*.

Structure or connection persistence occurs

Structures on the failed CF become persistent or structure connections become failed-persistent. This might happen for the following reasons:

- Structure rebuild failed.
 - An alternate CF did not exist.
 - Structure rebuild is not supported by the owning application.
- An application failure occurred.

If the application that owns a persistent structure or connection is restarted while the CF remains unavailable, its attempt to connect to the structure may not succeed (see [“A Review of the Concept of Persistence” on page 425](#)). A recovery bind is maintained, represented by the persistent structure or failed-persistent connection, to the instance of the structure that was in use at the time of the failure.

Until and unless connectivity to the CF containing that structure is restored, z/OS does not know whether the CF has truly failed (with the loss of structure data that implies), or z/OS has simply lost connectivity to the CF (with the implication that access to the data will be restored when connectivity to the CF is restored). This recovery bind will be maintained until one of the following events occurs:

- Connectivity to the CF is restored, and the structure is found to still exist (a loss and regain of connectivity scenario)
- Connectivity to the CF is restored, and the structure is found to no longer exist (a true CF failure scenario)
- Deletion of the structure or connection is forced (for example, via the SETXCF FORCE system command).

Thus, if you want to restart the application and have it allocate another instance of the structure in an alternate CF, you must first break the recovery bind to the old instance of the structure by deleting the persistent connections or the structure before structure allocation succeeds. See [“A Review of the Concept of Persistence” on page 425](#) before proceeding with structure or connection deletion.

Handling failed coupling facility structures

To determine which structures remain allocated on the failed CF, issue the `D XCF,CF,CFNAME=failed_cfname` command.

To delete the allocated structures do the following for each structure in the failed CF:

Display structure information using the `D XCF,STR,STRNM=strname` command.

The structure might have failed-persistent connections, or the structure itself might be persistent, or both.

- Deleting failed-persistent connections

Delete all failed-persistent connections with the following system command

```
SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL
```

Because the z/OS system does not have connectivity to the failed CF, the preceding command will be accepted with the following message:

```
IXC354I THE SETXCF FORCE REQUEST FOR CONNECTION
conname IN STRUCTURE strname WAS ACCEPTED:
REQUEST WILL BE PROCESSED ASYNCHRONOUSLY
```

Display the structure again: `DISPLAY XCF,STRUCTURE,STRNAME=strname`

The number of connections to the structure should now be zero.

- Deleting persistent structures

If the structure is persistent (DISPOSITION: KEEP in IXC360I) and the number of connections to the structure in message IXC360I is greater than zero, then delete the structure as described below.

If ACTIVE connectors exist (see connection state in IXC360I), invoke recovery procedures for the connector, or CANCEL the connector's address space to make the connector disconnect from the structure. Then, if the connector enters a failed-persistent state, force the connector using the SETXCF FORCE, CON.

Then, to delete the structure, issue the following system command: `SETXCF FORCE,STRUCTURE,STRNAME=strname`

Because the z/OS system does not have connectivity to the failed CF, the preceding command will be accepted with the following message:

```
IXC353I THE SETXCF FORCE REQUEST FOR STRUCTURE
strname WAS ACCEPTED:
REQUEST WILL BE PROCESSED ASYNCHRONOUSLY
```

Once all failed-persistent connections are forced or the structure is forced, the structure should enter an “in transition” state or “delete pending” state. If you display the structure again, message IXC360I should contain the following statements:

```
STRUCTURE IN TRANSITION
-----
REASON IN TRANSITION: CONNECT OR DISCONNECT IN PROGRESS
```

When the structure enters an *in transition* state, it means that the structure is *targeted for deletion* by CFRM. At this point, a new instance of the structure can be allocated. Depending on the application that owns the structure, the application might need to be restarted to allocate a new structure.

Note: When the structure is allocated in an alternate CF, the `DISPLAY XCF,STR,STRNAME=stname` command will produce information about both the new instance of the structure in the alternate CF and the *structure in transition* which is pending deletion from the failed CF. When the failed CF is eventually restored, XCF will resolve the *structure in transition* condition and this information will no longer be displayed.

Structure rebuild fails

If structure rebuild fails to complete, see [“Rebuild the structure” on page 454](#) for information on how to handle a rebuild hang.

Coupling facility storage constraints exist

If the alternate CF does not have enough capacity to handle the workload from the failed CF, consider removing less important structures from the alternate CF to allow the CF resources to drive your mission critical structures. See [“Planning for a Coupling Facility Shutdown” on page 430](#) for more information on handling insufficient CF capacity.

CASE 2 - An alternate coupling facility is not available

In this case, if there are structures critical to the operation of your sysplex, consider activating a CF in a backup logical partition. Follow the procedures for [“Adding a Coupling Facility” on page 429](#). Then proceed as in CASE 1.

Handling failing or disconnecting connection states

This information provides guidelines for recovering from structure connections that are hung in a FAILING or DISCONNECTING state. Unlike a connection state of ACTIVE or FAILED-PERSISTENT, a FAILING or DISCONNECTING state is a transitory state. When a connector disconnects from a structure, the disconnect process might place the connection into a DISCONNECTING or FAILING state. This is normal. However, a connection should not remain in the FAILING or DISCONNECTING state indefinitely.

Connections hung in a FAILING or DISCONNECTING state will likely be discovered when the associated connector job tries to reconnect to the structure and the reconnect (IXLCONN macro) request is rejected.

Determining the connection states

To determine the state of all connections for a given structure, issue the following command: `DISPLAY XCF,STRUCTURE,STRNAME=stname`

Message IXC360I will provide information about the structure and its connectors. Figure 84 on page 453 is an example of displaying information about an IRLM lock structure: `DISPLAY XCF,STRUCTURE,STRNAME=IRLMLOCK1`

```

IXC360I      12.23.28  DISPLAY XCF xxx
STRNAME:    IRLMLOCK1
STATUS:      ALLOCATED
.
.
/ (only relevant portions of message IXC360I
/   are shown here)
.

CONNECTION NAME  ID  VERSION  SYSNAME  JOBNAME  ASID  STATE
-----
IRLMGRP1$IRLA001 03  0003002F  SYS1     IRLMA    001D  ACTIVE
IRLMGRP1$IRLB002 02  0002008C  SYS2     IRLMB    0058  ACTIVE
IRLMGRP1$IRLC003 01  00010099  SYS3     IRLMC    003F  FAILING
IRLMGRP1$IRLC004 04  0004001A  SYS3     IRLMC    0180  DISCONNECTING

```

Figure 84. Example: Displaying information about an IRLM lock structure

In Figure 84 on page 453, connections 03 and 02 are ACTIVE (normal), and connections 01 and 04 are in transition.

Role of an active connector

Note that the connection that is hung in the FAILING or DISCONNECTING state is not the cause of the problem. The problem lies with one of the ACTIVE connectors. When a connector disconnects from a structure, all the other ACTIVE connectors to the structure must perform recovery for the FAILING or DISCONNECTING connection. If one or more of the ACTIVE connectors fails to perform and confirm recovery completion with z/OS, the FAILING or DISCONNECTING state cannot be exited. Hence, the problem is likely due to one of the ACTIVE connectors.

Steps for diagnosis and recovery

The following procedure allows you to identify the ACTIVE connector that is causing the problem, determine what recovery actions should be taken, collect documentation, and recover from the problem. You can wish to modify this procedure to suit your specific needs.

Identify the connector or systems that require recovery actions

By the time you find out that a connection is in a FAILING or DISCONNECTING state, it is likely that it had been in that state for a while. However, if the connection recently went into a FAILING or DISCONNECTING state (for example, you just recycled a subsystem), then you should ensure that enough time has elapsed to allow the FAIL and DISCONNECT events to be processed by the active connectors. Normally, this takes no more than a few minutes. IXL040E may be issued to identify a response that a connector has not provided.

- Determine if the FAILING or DISCONNECTING connection event is owed a response.

Issue the following command for each ACTIVE connection: DISPLAY
XCF, STRUCTURE, STRNAME=*strname*, CONNAME=*conname*

Message IXC360I will provide detailed information for the connector specified in the command. If the connector is owed a response (such as a response to a FAIL or DISCONNECT event), the detailed information might include the following:

```

CONNECTION ID(s) OWING A RESPONSE FOR THIS CONNECTOR:
id id id... id

```

The IDs owing a response to the FAILING or DISCONNECTING connection event identify the connector (job) that is not responding. Those jobs owing a response are candidates for recovery actions. If CONNECTION ID(s) OWING A RESPONSE was included in the IXC360I output, check the health of the job or system to see if there is anything obviously wrong with the job or system on which the job resides. If so, go to “Determine recovery actions” on page 454. Otherwise, continue to find information about the structure itself.

- Display all relevant information about the structure that might identify connectors or systems requiring recovery actions.

```
DISPLAY XCF,STRUCTURE,STRNAME=strnameCONNAME=ALL
```

For each ACTIVE connector (as displayed in IXC360I), check the health of the connector (job), and check the health of the system on which the ACTIVE connector resides.

- Issue commands that are processed by the ACTIVE connector. For example, if the connector is an IRLM job, issue: F IRLMx,STATUS F IRLMx,STATUS,ALLD

If the job owning the ACTIVE connector does not respond as normal, then it is a candidate for recovery actions.

- Issue commands on the system on which the ACTIVE connector resides. For example: D T

If the system does not respond as normal, then it is a candidate for recovery actions.

- Issue DISPLAY XCF,SYSPLEX,ALL to ensure that all systems are ACTIVE in the parallel sysplex.
- Issue DISPLAY R,L,CN=(ALL) to see if there are any outstanding events that might be holding up the processing of the FAILING or DISCONNECTING connection.
- Examine syslogs for signs of trouble such as ABENDs or DUMPs around the suspected time of the connector hang.

If a job owning an ACTIVE connector has experienced ABENDs or DUMPs, then it is a candidate for recovery actions. If the system has experienced problems such as ABENDs, DUMPs, SPINLOOPs, or other anomalies, then it is a candidate for recovery actions.

Determine recovery actions

The following topic describes recovery actions depending on what was found in the preceding steps:

- If something obvious was found, such as a critical WTOR that needed a response, respond as appropriate and verify that the FAILING or DISCONNECTING connection condition was resolved.
- If a job was identified as owing a response to a FAILING or DISCONNECTING connection event, it may be possible to force a response by attempting a structure rebuild. See [“Rebuild the structure” on page 454](#) below.
- If no job or system was identified as requiring a recovery action (everything appears to be normal), and if the structure supports rebuilding, continue with [“Rebuild the structure” on page 454](#). Otherwise, continue with [“Collect documentation” on page 455](#).

Rebuild the structure

Rebuild the structure that has the hung connectors by issuing the following command: SETXCF START,REBUILD,STRNAME=*strname* (See *z/OS MVS System Commands* for additional information.) If the structure does not support rebuild, go to [“Collect documentation” on page 455](#).

After the REBUILD command is issued, display the structure information by issuing the following command: DISPLAY XCF,STRUCTURE,STRNAME=*strname*

Examine the output of the command (IXC360I) and locate information about the REBUILD phase and the state of the structure connections. [Figure 85 on page 455](#) shows an example.

```

IXC360I 12.23.28 DISPLAY XCF xxx
STRNAME: IRLMLOCK1
STATUS: REASON SPECIFIED WITH REBUILD START:
        OPERATOR INITIATED
        REBUILD PHASE: QUIESCE
        .
        .
        / (only relevant portions of message IXC360I are
        / shown here)
        .

* ASTERISK DENOTES CONNECTOR WITH OUTSTANDING REBUILD RESPONSE

CONNECTION NAME  ID  VERSION  SYSNAME  JOBNAME  ASID  STATE
-----
* IRLMGRP1$IRLA001 03 0003002F SYS1    IRLMA    001D  ACTIVE
  IRLMGRP1$IRLB002 02 0002008C SYS2    IRLMB    0058  ACTIVE
  IRLMGRP1$IRLC003 01 00010099 SYS3    IRLMC    003F  FAILING
  IRLMGRP1$IRLC004 04 0004001A SYS3    IRLMC    0180  DISCONNECTING

```

Figure 85. Example: Output from `DISPLAY XCF, STRUCTURE, STRNAME` command

Determine if the REBUILD is “hung” by displaying the structure information again. Examine the REBUILD phase of the structure and the state of the connections. If the structure REBUILD is not progressing (for example, the structure is stuck in the REBUILD QUIESCE phase), look for connectors that have an outstanding REBUILD response (denoted by a * to the left of the connection name).

Note: Allow some time between issuing the display commands to ensure that the ACTIVE connectors have had a chance to process the REBUILD request. Structure REBUILD may take a few minutes. Display the structure a few times to verify that the REBUILD phase and the outstanding connector responses (*) are not changing. The REBUILD phase is displayed in message IXC360I. IXL040E is issued to identify a response that a connector has not provided in a timely manner.

If the REBUILD is not progressing and one or more of the ACTIVE connectors has an outstanding REBUILD response, the job associated with the connectors that are not responding to REBUILD (IRLMA in the above example), requires recovery actions. Continue with [“Collect documentation” on page 455](#).

Collect documentation

Dump the XCFAS address space and its data spaces on all systems containing connectors (ACTIVE or otherwise) to the structure with the hung connections. Use the `SDATA=(COUPLE,XESDATA)` options.

Dump the address spaces and data spaces of all ACTIVE connectors to the structure with the hung connections. In the example above, you would dump the XCFAS address space and data spaces on SYS1, SYS2, and SYS3, and IRLMA and IRLMB address spaces and their data spaces.

If a REBUILD was attempted and resulted in a REBUILD hang, and if the purpose of the REBUILD was simply to identify an ACTIVE connectors that owed a response to a connection that was in a FAILING or DISCONNECTING state, stop the REBUILD by issuing the following: `SETXCF STOP, REBUILD, STRNAME=strname`

Continue with [“Recover the Hung connectors” on page 455](#).

Recover the Hung connectors

In this step you will be shutting down one or more jobs, systems, or both to recover the hung connector(s). If it becomes necessary to shut down more than one job or system, consider shutting down one job or system at a time and then check to see if that action recovered the hung connector. If not, proceed to the next job or system shut down.

- If a job was identified as a candidate for recovery actions, shut down that job.
- If a system was identified as a candidate for recovery actions, shut down that system.

- If no jobs or systems were identified as candidates for recovery actions, shut down all jobs containing ACTIVE connectors to the structure with hung connectors.

After the shut down is complete, display the structure by issuing the following command:

```
DISPLAY XCF,STRUCTURE,STRNAME=strname
```

Examine the output of the command (IXC360I) and locate information about the structure connections. (In the following example, IRLMA on SYS1 was shut down.)

CONNECTION	NAME	ID	VERSION	SYSNAME	JOBNAME	ASID	STATE
IRLMGRP1\$IRLA001		03	0003002F	SYS1	IRLMA	001D	FAILED-PERSISTENT
IRLMGRP1\$IRLB002		02	0002008C	SYS2	IRLMB	0058	ACTIVE
IRLMGRP1\$IRLC003		01	00010099	SYS3	IRLMC	003F	FAILED-PERSISTENT

- If the FAILING and DISCONNECTING states have been resolved as indicated by the FAILING connector entering a FAILED-PERSISTENT or undefined state, and the DISCONNECTING connection becoming undefined, reinitialize the jobs that were shut down. Those jobs should restart and reconnect to the structure successfully.
- If the FAILING and DISCONNECTING states have not been resolved:
 - For the structure with the hung connectors, shut down all the remaining ACTIVE connectors. With all the active connectors shut down, the connections should all be in an undefined state or a FAILED-PERSISTENT state depending on the exploiter of the structure. Issue `DISPLAY XCF,STRUCTURE,STRNAME=strname` and verify that all connections have entered a FAILED-PERSISTENT or undefined state. Once verified, restart all jobs.
 - If all the connector jobs were shut down and the connections still have not entered a FAILED-PERSISTENT or undefined state, contact XCF Programming Support for additional assistance.

Best practices for updating a coupling facility

IBM recommends following the best practices documented herein when upgrading a coupling facility. These procedures for upgrading a coupling facility use the most current and simplified options to ensure a successful upgrade. These steps are intended to help you avoid unexpected delays or outages to applications that are using structures in coupling facilities. It is suggested that you follow all of the steps in the procedures. When adhered to, the procedures are designed to maximize availability and reliability. The best practice procedures contained assume that you want to keep the sysplex operational across the upgrade.

You can follow these procedures either in this book, or in a workflow in z/OSMF. For the z/OSMF workflows, see the following link:

```
https://github.com/IBM/IBM-Z-zOS/tree/master/zOS-Workflow/XCF%20Workflows%20-%20Upgrading%20a%20CF
```

Note: In the context of this section, a "processor" is defined as a physical collection of hardware that consists of main storage, one or more central processors, timers, and channels, on which a coupling facility image resides.

The first procedure, "Push/pull of a coupling facility or POR of a processor with a coupling facility" on page 457, should be used for either of the following situations:

- There is a "push/pull" of a CPC on which a coupling facility image resides
- There is to be a power-on reset, POR, of the CPC on which the coupling facility image resides and there is a physical or logical change to the coupling facility.

A "push/pull" is defined as the replacement of one CPC containing a CF image with another CPC that contains a replacement CF image. Physical changes to a coupling facility include the number of links, the control unit number, where the links physically connect. The logical change of a coupling facility refers to the coupling facility definition in the CFRM policy.

The next procedure, [“POR of a CPC with a Coupling Facility with no Physical or Logical Changes to the CF”](#) on page 461, also pertains to a POR of a CPC on which a coupling facility image resides. However, this procedure applies to the specific situation where there is to be a POR of a CPC on which a coupling facility resides and across the POR, there are no physical or logical changes to the coupling facility.

Finally, [“Disruptive CF Upgrade”](#) on page 464 documents the procedure for a disruptive coupling facility upgrade. The disruptive coupling facility upgrade procedure should be conformed to anytime the coupling facility image must be reactivated but the CPC on which the coupling facility resides is not going to be PORd. Most CFCC maintenance applications can be achieved concurrently. That is, the CF image does not need to be reactivated to pick up the new service level of CFCC. The reasons a CF image may need to be reactivated include: the rare disruptive coupling facility code change, change to CFCC image storage requirements, and activating a new Coupling Facility Release level that is typically delivered with an IBM Z driver upgrade.

Caution: Failure to adhere to the recommended best practices may result in unexpected delays or outages to applications that use structures in coupling facilities. Also, technical difficulties trying to obtain access to the new coupling facility may occur if any of the steps or sequence of steps is not properly followed.

"Push/pull" of a coupling facility or POR of a processor with a coupling facility

Use the procedure shown in [Table 37 on page 457](#) for **either** of the following situations:

- a CF is going to move onto a new CPC
- the CPC on which the coupling facility resides is going to be PORd **and** there are physical or logical changes to the CF

Note: If the CPC is going to be PORd and there are *no physical or logical changes* to the CF, use [“POR of a CPC with a Coupling Facility with no Physical or Logical Changes to the CF”](#) on page 461 .

Table 37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides		
Step	Command	Reason
0	<p>Create new CFRM policy distinct from the currently active policy with the new CF definition and updated structure definitions based on CFSizer or SIZER.</p> <p>Alternatively, update the current policy "in place" to avoid changing the name of the policy.</p>	<p>This step can be done ahead of time to minimize net down time. For the "push/pull", the node descriptor for the CF must be updated (machine type, serial number, partition number, etc.). Appropriate structure sizes for the new CFCC level can be obtained from Coupling Facility sizer (www.ibm.com/support/docview.wss?uid=isg3T1027062) or CFSizer Alternate Sizing Techniques (www.ibm.com/support/docview.wss?uid=isg3T1025939).</p> <p>Updated structure sizes are necessary to ensure that a performance problem does not occur due to a structure space constraint. Also, the updated structure sizes will ensure that there is no issue restarting an application in the future due to invalid structure sizes. The “resizing” of structures is required when upgrading to a new CFCC level, such as, CFCC 16 (z10) to CFCC 17 (z196).</p>
1	<ol style="list-style-type: none"> 1. Quiesce all work on z/OS images that reside on the CPC being PORd or removed. 2. Remove z/OS images from the sysplex by issuing <pre>V XCF,sysname,OFFLINE</pre>	
2	Reassign STP roles, as needed.	<p>Ensure CTN will not be unexpectedly disrupted when the "old" CPC where the CF image resides is removed.</p> <p>For more information, see IBM White Paper 101833, Important Considerations for STP server role assignments (www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101833).</p>

Table 37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides (continued)		
Step	Command	Reason
3	SETXCF START, MAINTMODE, CFNAME= <i>cfname</i>	Place the CF to be upgraded in maintenance mode so that no new structures will be allocated on the CF.
4	D XCF, REALLOCATE, TEST (available z/OS Version 1 Release 12)	Preview the results of REALLOCATE and evaluate any exceptions. If a severe problem is detected, remove <i>cfname</i> from MAINTMODE and address the problem. Then, go back to Step 3 .
5	SETXCF START, REALLOCATE	Issue REALLOCATE to have XCF assess each structure and relocate, as appropriate. XCF will seek to move the structures off the CF in MAINTMODE.
6a	D XCF, CF, CFNAME= <i>cfname</i>	Determine if any structures remain in the CF that is being removed from service. If no structures remain in the CF being removed, go to Step 8 . If structures remain in the CF being removed, follow Step 6b and Step 7 .
6b	D XCF, REALLOCATE, REPORT (available z/OS Version 1 Release 12)	If any structure did not move, review the REALLOCATE REPORT output and address any errors.
7	If any structures remain in CF <i>cfname</i> , issue the following commands: SETXCF START, REBUILD, STRNAME= <i>strname</i> , LOC=OTHER SETXCF STOP, REBUILD, DUPLEX, STRNAME= <i>strname</i> , KEEP=NEW OLD D XCF, CF, CFNAME= <i>cfname</i>	Move any structures that remain in <i>cfname</i> . Application-specific protocols may be needed to move structures. Then, verify that no structures remain on the coupling facility about to be upgraded. For details about moving structures out of the CF, see “Removing structures from a coupling facility for shutdown” on page 437 and “Removing a structure from a coupling facility” on page 438.
8	VARY PATH(CFNAME,xx,CFNAME,yy,etc), OFFLINE, UNCOND	VARY the paths to the CF offline. The VARY command must be issued from all systems in the sysplex; path numbers may be different for each system. Taking the paths offline and then recycling is cleaner for z/OS to handle than not taking the paths offline. The hot unplug or TOGGLE is received as an “error” by z/OS. To obtain the list of CHPIDs, issue D CF from each system. The list of CHPIDs is helpful when bringing the paths back online. The VARY command is used rather than the CONFIG command because it is possible that use of the links by STP would cause the CONFIG command to be rejected. Using the VARY command affects only the logical path state for coupling facility communication and does not affect (nor is it affected by) the STP network; STP can still utilize the links to maintain the timing network. When the paths are VARYd OFFLINE, the D CF, CFNAME= <i>cfname</i> command shows the paths as being logically offline, but physically online.
9	RO *ALL, D CF, CFNAME= <i>cfname</i> D XCF, CF, CFNAME= <i>cfname</i>	Verify that no system has an active path to CF. At this point, the paths will indicate logically OFFLINE. Re-verify that no structures in the CF and no systems have connectivity to CF.

Table 37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides (continued)

Step	Command	Reason
10	<p>From the CF OPERMSG console, issue:</p> <pre>SHUTDOWN</pre>	<p>The SHUTDOWN command is recommended to avoid the potential error of DEACTIVATING the coupling facility that has all of the structures allocated in it. The SHUTDOWN command will not complete if structures are present in the coupling facility.</p> <p>The following scenario shows the message sequence for the SHUTDOWN command when structures are still allocated in the CF.</p> <pre>=> shutdown CF0090A Do you really want to shut down the Coupling Facility? (YES/NO) => yes CF0093A There are structures present in the CF. SHUTDOWN canceled.</pre> <p>After the SHUTDOWN command completes successfully, the CF LPAR will become “not operating.”</p> <p>If structures are still allocated in the CF, verify that the correct CF was selected for SHUTDOWN. If the correct CF was selected, consider reviewing the structures still allocated in the CF and seek to move them out of the CF. At this point, the paths must be VARYd back ONLINE to allow structures to rebuild out of the CF. VARY the paths ONLINE and go back to Step 6a. If VARYing the paths ONLINE and moving the structures is not desired (that is, deleting the structures is acceptable), then the CF image may be DEACTIVATED.</p>
11	Remove the CPC about to be PORd or removed from the CTN.	Removal from the CTN will allow the CONFIG command(s) in Step 12 to be accepted.
12	CONFIG CHP(xx,yy...),OFFLINE,UNCOND	<p>Configure the links to the CF OFFLINE. D CF, CFNAME=cfname and D M=CHP will show the links as physically OFFLINE. Recall that paths were already logically OFFLINE from the VARY OFFLINE issued previously.</p> <p>The CONFIG command is used (in addition to the VARY PATH command) to ensure that z/OS can recognize all hardware changes.</p>
13	Bring in the “new machine”.	Remove the “old” machine and bring in the “new” machine or, POR the CPC, depending on the scenario.
14	<p>ACTIVATE parms,SOFT=VALIDATE</p> <p>On the Nth partition, issue:</p> <pre>ACTIVATE parms,FORCE</pre>	<p>If making changes to CF elements (CF control units or CF channel paths) in the I/O configuration, ensure that SOFT=VALIDATE is specified in the <i>parms</i> on the ACTIVATE system command. SOFT=VALIDATE is a requirement on all N-1 partitions when changes to CF elements are made.</p> <p>When SOFT=VALIDATE is specified, HCD builds the change control blocks, CCBs, for hardware changes, as well as, software changes. SOFT=VALIDATE also ensures there is sufficient hardware system area (HSA) space to accommodate the changes.</p> <p>On the Nth partition, activate the <i>parms</i>.</p> <p>For more information, refer to "Activate Command Parameters" in z/OS MVS System Commands.</p>

Table 37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides (continued)

Step	Command	Reason
15	<pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=new_policy</pre> or <pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=mod_active_policy</pre>	<p>Activate the new CFRM policy with the definition for the new CF and updated structure sizes per CFSizer or SIZER output. Recall, the new CF definitions is only required for a "push/pull" scenario. The CF definition change will take effect immediately. Structure size changes will be pending after the new or modified policy is activated.</p> <p>From a CFRM perspective, the previous definition was removed and a new CF definition was added. The new CF definition will not be in MAINTMODE even if the CFNAME is the same.</p>
16	<pre>SETXCF START,MAINTMODE,CFNAME=cfname</pre>	Place the newly-defined CF in MAINTMODE to prevent DUPLEX(ENABLED) structures from immediately moving into the CF when connectivity is established.
17	<ol style="list-style-type: none"> 1. Add the CPC that was PORd back into the CTN or add the new CPC to the CTN. 2. Activate the CF partition. 3. Reassign STP roles as necessary. 	<ol style="list-style-type: none"> 1. The CPC must be added to the CTN to be properly synchronized with the other CPCs. 2. CF must be activated to allow the system to connect and use the image. 3. Ensure STP is in the configuration you want to use. <p>For more information, see IBM White Paper 101833, Important Considerations for STP server role assignments (www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101833).</p>
18	<pre>CONFIG CHP(xx,yy,etc),ONLINE</pre> <pre>V PATH(CFname,xx,CFname,yy,etc),ONLINE</pre>	<p>Configure CHPIDs online. The configuration commands must be issued from all systems in the sysplex; CHPID numbers may be different for each system.</p> <p>If the CHPIDs numbers have not changed on z/OS across the "push/pull", use the links noted in Step 8.</p> <p>Message IXC517I will be issued indicating that XCF has connectivity to the CF.</p> <p>Note: If a policy change was done, the logical state of the paths is their default state (ONLINE), so there is no need to use the VARY command to bring the paths logically online. However, if the same links are to be used and there was no change to the CF definition in the CFRM policy, the logical state will be OFFLINE. Thus, the paths would need to be VARYd ONLINE from all of the z/OS images that remained active.</p>
19	<pre>D XCF,CF,CFNAME=cfname</pre> <pre>RO *ALL,D CF,CFNAME=cfname</pre>	<p>Verify XCF has the proper CF definition. In particular, if a "push/pull" is being done ensure the serial number for the new CF. <code>D XCF,CF,CFNAME=cfname</code> indicates that the CF definition XCF logically knows about. <code>D CF,CFNAME=cfname</code> contains the physical information for the CF the image is connected to. The serial numbers must match for XCF to be able to use the CF.</p> <p>Verify that all systems have connectivity to the new CF and all the paths are ONLINE. Also, verify that CF-to-CF links are available.</p>
20	<pre>SETXCF STOP,MAINTMODE,CFNAME=cfname</pre>	Take the newly defined CF out of MAINTMODE to allow structure allocations in the new CF.
21	<pre>SETXCF START,REALLOCATE</pre>	Relocate structures to the desired CFs. XCF will seek to resolve the pending policy changes.
22	<pre>D XCF,REALLOCATE,REPORT</pre> (available on z/OS Version 1 Release 12)	Verify that all structures reside in the desired CFs. Correct any errors noted by the report.

Table 37. Steps: move a coupling facility for a push/pull operation or a POR of the processor where the CF resides (continued)		
Step	Command	Reason
23	<p>Verify that structures reside in the preferred CFs. Review output from z/OS Health Check XCF_CF_STR_PREFLIST.</p> <p>Use the following SETXCF REBUILD commands, as necessary, to move structures.</p> <pre>SETXCF START,REBUILD,STRNAME=stname,LOC=OTHER</pre> <pre>SETXCF</pre> <pre>STOP,REBUILD,DUPLEX,STRNAME=stname,KEEP=NEW OLD</pre> <p>In cases where additional CFRM policy updates are warranted, (such as changing a duplex CF site preference or a CF site specification), the SETXCF START,REALLOCATE command may be used again once the new policy is started.</p>	<p>REALLOCATE processing may complete with 0 exceptions because XCF / XES placed the structure in most desirable CFs, which may not coincide with the order of the CFs in the PREFLIST or the desired CF site preference for duplexed structures.</p> <p>Investigate structures that were not placed in the desired CF PREFLIST order, taking into account any CF site preferences for duplexed structures.</p> <p>Assess the reason for the structure placement and relocated structures to desired CFs, as needed.</p>
24	IPL z/OS images that reside on the CPC that was PORd or reside on the new CPC.	

POR of a CPC with a Coupling Facility with no Physical or Logical Changes to the CF

Use the procedure in Table 38 on page 461 when the CPC on which the coupling facility resides is going to be PORd and there are **no** physical or logical changes to the CF.

Note: If physical or logical changes to the coupling facility are to be made, use the procedure described in “Push/pull of a coupling facility or POR of a processor with a coupling facility” on page 457.

Table 38. Steps: POR of a CPC with a coupling facility with no physical or logical changes to the CF		
Step	Command	Reason
0	<p>Create new CFRM policy distinct from the currently active policy with the updated structure definitions based on CFSizer or SIZER.</p> <p>Alternatively, update the current policy "in place" to avoid changing the name of the policy.</p>	<p>This step can be done ahead of time to minimize net down time. Appropriate structure sizes for the new CFCC level can be obtained from Coupling Facility sizer (www.ibm.com/support/docview.wss?uid=isg3T1027062) or CFSizer Alternate Sizing Techniques (www.ibm.com/support/docview.wss?uid=isg3T1025939).</p> <p>Updated structure sizes are necessary to ensure that a performance problem does not occur due to a structure space constraint. Also, the updated structure sizes will ensure that there is no issue restarting an application in the future due to invalid structure sizes.</p> <p>This step is required for CFCC Level changes. This step is desirable if the sizes were not adjusted across the last CFCC Level upgrade or if the usage of the structures has changed significantly.</p> <p>Caution: This procedure assumes the CF links, control unit, and CFRM definition are not changing. If the links, control unit, or CFRM definition change, the CF may become accessible earlier than planned.</p>
1	<ol style="list-style-type: none"> 1. Quiesce all work on z/OS images that reside on the CPC being PORd or removed. 2. Issue the following command to remove those z/OS images from the sysplex: <pre>V XCF,sysname,OFFLINE</pre>	

Table 38. Steps: POR of a CPC with a coupling facility with no physical or logical changes to the CF (continued)

Step	Command	Reason
2	Reassign STP roles, as needed.	Ensure that CTN will not be disrupted unexpectedly when the "old" CPC where the CF image resides is removed. For more information, see IBM White Paper 101833, Important Considerations for STP server role assignments (www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101833).
3	SETXCF START,MAINTMODE,CFNAME= <i>cfname</i>	Place the CF to be upgraded in maintenance mode, so that no new structures will be allocated on the CF.
4	D XCF,REALLOCATE,TEST (available z/OS Version 1 Release 12)	Preview the results of REALLOCATE. Evaluate any exceptions. If a severe problem is detected, remove <i>cfname</i> from MAINTMODE and address the problem. Then, go back to Step 3.
5	SETXCF START,REALLOCATE	Issue REALLOCATE to have XCF assess each structure and relocate, as appropriate. XCF will seek to move the structures off the CF in MAINTMODE.
6a	D XCF,CF,CFNAME= <i>cfname</i>	Determine if any structures remain in the CF that is being removed from service. If no structures remain in the CF being removed, go to Step 8. If structures remain in the CF being removed, follow Step 6b and Step 7.
6b	D XCF,REALLOCATE,REPORT (available z/OS Version 1 Release 12)	If a structure did not move, review the REALLOCATE REPORT output and address any errors.
7	If structures remain in CF <i>cfname</i> , issue the following commands: SETXCF START,REBUILD,STRNAME= <i>strname</i> ,LOC=OTHER SETXCF STOP,REBUILD,DUPLEX,STRNAME= <i>strname</i> ,KEEP=NEW OLD D XCF,CF,CFNAME= <i>cfname</i>	Move any structures that remain in <i>cfname</i> . Application-specific protocols may be needed to move structures. Then, verify that no structures remain on the coupling facility about to be upgraded. For details about moving structures out of the CF, see "Removing structures from a coupling facility for shutdown" on page 437 and "Removing a structure from a coupling facility" on page 438.
8	VARY PATH(CFNAME,xx,CFNAME,yy,etc),OFFLINE,UNCOND	VARY the paths to the CF offline. The VARY command must be issued from all systems in the sysplex; path numbers may be different for each system. Taking the paths offline and then recycling is cleaner for z/OS to handle than not taking the paths offline. The hot unplug or TOGGLE is received as an "error" by z/OS. To obtain the list of CHPIDs, issue D CF from each system. The list of CHPIDs will be helpful when bringing the paths back online. The VARY command is used rather than the CONFIG command because it is possible that use of the links by STP would cause the CONFIG command to be rejected. Using the VARY command affects only the logical path state for coupling facility communication and does not affect (nor is it affected by) the STP network; STP can still use the links to maintain the timing network. When the paths are VARYd OFFLINE, the D CF,CFNAME= <i>cfname</i> command shows the paths as being logically offline, but physically online.
9	RO *ALL,D CF,CFNAME= <i>cfname</i> D XCF,CF,CFNAME= <i>cfname</i>	Verify that no system has an active path to CF. At this point, the paths will indicate logically OFFLINE. Re-verify that no structures in the CF and no systems have connectivity to CF.

Table 38. Steps: POR of a CPC with a coupling facility with no physical or logical changes to the CF (continued)

Step	Command	Reason
10	From the CF OPERMSG console, issue: SHUTDOWN	<p>The SHUTDOWN command is recommended to avoid the potential error of DEACTIVATING the coupling facility that has all of the structures allocated in it. The SHUTDOWN command will not complete if structures are present in the coupling facility.</p> <p>The following scenario shows the message sequence for the SHUTDOWN command when structures are still allocated in the CF:</p> <pre>=> shutdown CF0090A Do you really want to shut down the Coupling Facility? (YES/NO) => yes CF0093A There are structures present in the CF. SHUTDOWN canceled.</pre> <p>After the SHUTDOWN command completes successfully, the CF LPAR will become “not operating.”</p> <p>If structures are still allocated in the CF, verify that the correct CF was selected for SHUTDOWN. If the correct CF was selected, consider reviewing the structures still allocated in the CF and seek to move them out of the CF. At this point, the paths must be VARYd back ONLINE to allow structures to rebuild out of the CF. VARY the paths ONLINE and go back to Step 7. If VARYing the paths ONLINE and moving the structures is not desired (that is, deleting the structures has been deemed acceptable), the CF image may be DEACTIVATED.</p>
11	Perform Power On Reset.	
12	Activate the CF partition.	CF must be activated to allow the system to connect and utilize the image.
13	Reassign STP roles as necessary.	<p>Ensure STP is in the desired configuration.</p> <p>For more information, see IBM White Paper 101833, Important Considerations for STP server role assignments (www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101833).</p>
14	V PATH(CFname,xx,CFname,yy,etc) ,ONLINE	Because there was no change to the CF definition in the CFRM policy and the same links and control unit are to be used to access the CF, the logical state of the paths will be OFFLINE. Thus, the paths need to be VARYd ONLINE from all of the z/OS images that remained active. The list of paths made in Step 8 can be referenced.
15	<pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=new_policy</pre> <p>or</p> <pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=mod_active_policy</pre>	<p>Activate the new CFRM policy with the definitions for the updated structure sizes, per CFSizer or SIZER output. Structure size changes will be pending after the new or modified policy is activated.</p> <p>Bypass this step, if no policy was created or modified in Step 0.</p>
16	SETXCF STOP,MAINTMODE,CFNAME=cfname	Take the newly-defined CF out of MAINTMODE to allow structure allocations in the new CF.
17	SETXCF START,REALLOCATE	Relocate structures to the desired CFs. XCF will seek to resolve the pending policy changes.
18	D XCF,REALLOCATE,REPORT (available on z/OS Version 1 Release 12)	Verify that all structures reside in desired CFs. Correct any errors noted by the report.

Table 38. Steps: POR of a CPC with a coupling facility with no physical or logical changes to the CF (continued)		
Step	Command	Reason
19	<p>Verify that structures reside in preferred CFs. Review output from z/OS Health Check XCF_CF_STR_PREFLIST.</p> <p>Issue the following SETXCF REBUILD commands, as necessary, to move structures.</p> <pre>SETXCF START,REBUILD,STRNAME=strname,LOC=OTHER SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=NEW OLD</pre>	REALLOCATE processing may complete with 0 exceptions because XCF / XES placed the structure in the most desirable CFs; this may not coincide with the order of the CFs in the PREFLIST. Investigate structures that were not placed in the CF that is noted first in the PREFLIST. Assess the reason for the structure placement and relocated structures to desired CFs as needed.
20	IPL z/OS images that reside on the CPC that was PORd.	

Disruptive CF Upgrade

Use the procedure in Table 39 on page 464 when the coupling facility image must be reactivated, but the CPC on which the coupling facility resides will remain operational across the reactivation process. This procedure also assumes the CF links, CF control unit, and CF definition in the CFRM policy are **not** changing.

Table 39. Steps: move a disrupted coupling facility		
Step	Command	Reason
0	<p>Create new CFRM policy distinct from the currently active policy with the updated structure definitions based on CFSizer or SIZER.</p> <p>Alternatively, update the current policy "in place" to avoid changing the name of the policy.</p>	<p>This step can be done ahead of time to minimize net down time. Appropriate structure sizes for the new CFCC level can be obtained from Coupling Facility sizer (www.ibm.com/support/docview.wss?uid=isg3T1027062) or CFSizer Alternate Sizing Techniques (www.ibm.com/support/docview.wss?uid=isg3T1025939).</p> <p>Updated structure sizes are necessary to ensure that a performance problem does not occur due to a structure space constraint. Also, the updated structure sizes will ensure that there is no issue restarting an application in the future due to invalid structure sizes.</p> <p>This step is required for CFCC Level changes. This step is desirable if the sizes were not adjusted across the last CFCC Level upgrade or if the usage of the structures has changed significantly.</p> <p>Caution: This procedure assumes the CF links, control unit, and CFRM definition are not changing. If the links, control unit, or CFRM definition change, the CF may become accessible earlier than planned.</p>
1	SETXCF START,MAINTMODE,CFNAME=cfname	Place the CF to be upgraded in maintenance mode so that no new structures will be allocated on the CF.
2	D XCF,REALLOCATE,TEST (available z/OS Version 1 Release 12)	Preview the results of REALLOCATE. Evaluate any exceptions. If a severe problem is detected, remove <i>cfname</i> from MAINTMODE and address the problem. Then, go back to Step 1 .
3	SETXCF START,REALLOCATE	Issue REALLOCATE to have XCF assess each structure and relocate, as appropriate. XCF will seek to move the structures off the CF in MAINTMODE.
4a	D XCF,CF,CFNAME=cfname	<p>Determine if any structures remain in the CF that is being removed from service.</p> <p>If no structures remain in the CF being removed, go to Step 6. If structures remain in the CF that is being removed, follow Step 4b and Step 5.</p>
4b	D XCF,REALLOCATE,REPORT (available z/OS Version 1 Release 12)	If a structure did not move, review the REALLOCATE REPORT output and address any errors.

Table 39. Steps: move a disrupted coupling facility (continued)

Step	Command	Reason
5	<p>If structures remain in CF <i>cfname</i>, issue the following commands:</p> <pre>SETXCF START,REBUILD,STRNAME=<i>strname</i>,LOC=OTHER SETXCF STOP,REBUILD,DUPLEX,STRNAME=<i>strname</i>,KEEP=NEW OLD D XCF,CF,CFNAME=<i>cfname</i></pre>	<p>Move any structures that remain in <i>cfname</i>. Application-specific protocols may be needed to move structures. Then, verify that no structures remain on the coupling facility about to be upgraded.</p> <p>For details about moving structures out of the CF, see “Removing structures from a coupling facility for shutdown” on page 437 and “Removing a structure from a coupling facility” on page 438.</p>
6	<p>On each system, issue:</p> <pre>VARY PATH(CFname,xx,CFname,yy,etc),OFFLINE,UNCOND</pre>	<p>VARY the paths to the CF offline. The VARY command must be issued from all systems in the sysplex; path numbers may be different for each system. Taking the paths offline and then recycling is cleaner for z/OS to handle than not taking the paths offline. The hot unplug or TOGGLE is received as an “error” by z/OS.</p> <p>To obtain the list of CHPIDs, issue D CF from each system. The list of CHPIDs is helpful when bringing the paths back online.</p> <p>The VARY command is used rather than the CONFIG command because it is possible that use of the links by STP would cause the CONFIG command to be rejected. Using the VARY command affects only the logical path state for coupling facility communication and does not affect (nor is it affected by) the STP network; STP can still use the links to maintain the timing network.</p> <p>When the paths are VARYd OFFLINE, the D CF,CFNAME=<i>cfname</i> command shows the paths as being logically offline, but physically online.</p>
7	<pre>RO *ALL,D CF,CFNAME=<i>cfname</i> D XCF,CF,CFNAME=<i>cfname</i></pre>	<p>Verify that no system has an active path to CF. At this point, the paths will indicate logically OFFLINE. Reverify that no structures in the CF and no systems have connectivity to CF.</p>
8	<ol style="list-style-type: none"> From the CF OPERMSG console, issue: SHUTDOWN. After the SHUTDOWN completes, ACTIVATE the CF. <p>Note: Do not perform a POR the CPC. If there is to be a POR of the CPC, use procedure “Push/pull of a coupling facility or POR of a processor with a coupling facility” on page 457.</p>	<p>The SHUTDOWN command is recommended to avoid the potential error of DEACTIVATING the coupling facility that has all of the structures allocated in it. The SHUTDOWN command will not complete if structures are present in the coupling facility.</p> <p>The following scenario shows the message sequence for the SHUTDOWN command when structures are still allocated in the CF:</p> <pre>=> shutdown CF0090A Do you really want to shut down the Coupling Facility? (YES/NO) => yes CF0093A There are structures present in the CF. SHUTDOWN canceled.</pre> <p>After the SHUTDOWN command completes successfully, the CF LPAR will become “not operating.”</p> <p>If structures are still allocated in the CF, verify that the correct CF was selected for SHUTDOWN. If the correct CF was selected, consider reviewing the structures still allocated in the CF and seek to move them out of the CF. At this point, the paths must be VARYd back ONLINE to allow structures to rebuild out of the CF. VARY the paths ONLINE and go back to Step 5. If you do not want to VARY the paths ONLINE and move the structures (that is, deleting the structures is acceptable), the CF image may be DEACTIVATED.</p>

Table 39. Steps: move a disrupted coupling facility (continued)

Step	Command	Reason
9	<pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=new_policy</pre> or <pre>SETXCF START,POLICY,TYPE=CFRM,POLNAME=mod_active_policy</pre>	Activate the new CFRM policy with the definitions for the updated structure sizes, per CFSizer or SIZER output. Structure size changes will be pending after the new or modified policy is activated.
10	<p>On each system, issue:</p> <pre>VARY PATH(CFname,xx,CFname,yy,etc),ONLINE</pre>	VARY paths online. The command must be issued from all systems in the sysplex; path numbers may be different for each system. To obtain the list of CHPIDs, issue D CF from each system or use the list from Step 6 .
11	<pre>D XCF,CF,CFNAME=cfname</pre> <pre>RO *ALL,D CF,CFNAME=cfname</pre>	If any changes were made to the CF definitions in the CFRM policy, verify that XCF is connected to the correct CF, including the serial number for the new CF. D XCF,CF,CFNAME=cfname indicates the CF definition that XCF logically knows about. D CF,CFNAME=cfname contains the physical information for the CF the image is connected to. The serial numbers noted in the D XCF and D CF commands for each CF must match. Verify that all systems have connectivity to the new CF and all the paths are ONLINE. Also, verify that CF to CF links are available.
12	<pre>SETXCF STOP,MAINTMODE,CFNAME=cfname</pre>	Take the newly-defined CF out of MAINTMODE to allow structure allocations in the new CF.
13	<pre>SETXCF START,REALLOCATE</pre>	Relocate structures to the desired CFs. XCF will seek to resolve the changes pending.
14	<pre>D XCF,REALLOCATE,REPORT</pre> (available on z/OS Version 1 Release 12)	Verify that all structures reside in desired CFs. Correct any errors noted by the report.
15	<p>Verify that structures reside in preferred CFs. Review output from z/OS Health Check XCF_CF_STR_PREFLIST.</p> <p>Issue the following SETXCF REBUILD commands, as necessary, to move structures.</p> <pre>SETXCF START,REBUILD,STRNAME=strname,LOC=OTHER</pre> <pre>SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=NEW OLD</pre>	REALLOCATE processing may complete with 0 exceptions because XCF / XES placed the structure in the most desirable CFs; this may not coincide with the order of the CFs in the PREFLIST. Investigate structures that were not placed in the CF that is noted first in the PREFLIST. Assess the reason for the structure placement and relocated structures to the desired CFs, as needed.

Chapter 18. Using automatic tape switching (ATS STAR)

When you configure your tape devices in a sysplex, you must consider whether to have all your tape drives on one system, or whether you will spread them out. You can share tape devices across many systems, both with and without the need for manual intervention by the operator. IBM offers the following ways for many systems to use the same tape devices. A tape device can be:

- A **dedicated tape device** is varied online to one system at a time. For a second system to use that same device, an operator issues VARY commands (first VARY OFFLINE, then VARY ONLINE) to make the device available to the second system.
- An **automatically switchable tape device** can be online to more than one system at a time. For one system to use an automatically switchable tape device, then another system to use the same device, an operator does not have to issue any VARY commands. When the system selects that device for allocation to a job step, the device is at once both assigned to the specific system and allocated to the job step. Automatically switchable tape devices require that the systems in the sysplex communicate with each other.

Automatic tape switching (that is, using automatically switchable tape devices) can benefit a sysplex by:

- Requiring fewer tape devices in a sysplex

By using automatically switchable tape devices, you can reduce the overall number of tape devices needed by a sysplex. With tape devices easily available to multiple systems, a sysplex can use its tape resources more efficiently.

- Reducing operations cost

Automatic tape switching can reduce operating costs by managing tape drives at the sysplex level and eliminating costly operator intervention.

For a system to consider a device automatically switchable, these conditions must be true:

- The device must be defined as automatically switchable.

You can define the device as automatically switchable through HCD.

You can issue the VARY *devnum* AS ONLINE command when the device is offline, and then VARY it online.

- The device must be varied online through the VARY ONLINE command or the IEEVARYD macro.

A system not participating in automatic tape switching but connected to a device that is defined as automatically switchable can vary that device online. During the time the device is online to the nonparticipating system, the device is not available to the participating systems. After the device is varied offline, it becomes available to participating systems.

For general information about automatic tape switching, see [z/OS HCD Planning](#).

Setting up automatic tape switching

Prior to z/OS Release 2, automatic tape switching was accomplished by using the IEFAUTOS coupling facility structure. Systems in a sysplex stored the status of online automatically switchable tape devices in IEFAUTOS.

With z/OS R2, the ATS STAR design improved the availability and system management characteristics of the prior automatic tape switching function. The ATS STAR design dropped the use of the IEFAUTOS structure and instead used global resource serialization and XCF services to maintain serialization when allocating shared tape devices. To maximize the performance of the ATS STAR function, it is strongly

recommended that you use the global resource serialization Star configuration, rather than the Ring configuration.

Recommendations for automatic tape switching

IBM recommends the following when setting up automatic tape switching:

- For ease of tape device management, IBM recommends that you use the same device number on all systems to represent an automatically switchable device. If an automatically switchable device is a 3480 or a 3490 without the 'Read Configuration Data Capable' function, you must use the same device number.
- To support automatic tape switching, you also need to update the GRSRNLxx member of parmlib by adding the following system inclusion RNL:

```
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSZVOLS)
```

This specification prevents a system from holding a tape device while it waits for a mount of a volume that is in use by another system in the sysplex.

- For ease of tape volume management, autoswitchable devices should be online to all systems in the sysplex. Systems that do not have an autoswitchable device online cannot detect any volume that is mounted on that device. If a system leaves a volume mounted on a device that is not online to all systems through the use of DISP=(,PASS) or DISP=(,RETAIN), systems without the device online cannot detect that the volume is mounted and cannot dismount it. In this situation, it may be necessary to manually dismount the volume.

Appendix A. Accessibility

Accessible publications for this product are offered through [IBM Documentation for z/OS \(www.ibm.com/docs/en/zos\)](http://www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS documentation see [How to Send Feedback to IBM](#) to leave documentation feedback.

Notices

This information was developed for products and services that are offered in the USA 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 grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America*

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.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

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

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

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:

*IBM Corporation
Site Counsel
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 document 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.

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

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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

COPYRIGHT LICENSE:

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at ibm.com/privacy and IBM's Online Privacy Statement at ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, 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 \(www.ibm.com/software/support/systemsz/lifecycle\)](http://www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.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 on the Web at [Copyright and Trademark information \(www.ibm.com/legal/copytrade.shtml\)](http://www.ibm.com/legal/copytrade.shtml).

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Glossary

This glossary defines technical terms and abbreviations used in sysplex documentation.

A

Automatic restart management

Automatic restart management is an MVS recovery function that improves the availability of batch jobs and started tasks. When a job fails, or the system on which it is running unexpectedly fails, MVS can restart the job without operator intervention.

C

central processor (CP)

The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

central processor complex (CPC)

A physical collection of hardware that consists of main storage, one or more central processors, timers, and channels.

CFRM

Coupling facility resource management.

channel-to-channel (CTC)

Refers to the communication (transfer of data) between programs on opposite sides of a channel-to-channel adapter (CTCA).

couple data set

A data set that is created through the XCF couple data set format utility and, depending on its designated type, is shared by some or all of the MVS systems in a sysplex. See also *sysplex couple data set*.

coupling facility

A special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex.

coupling facility channel

A high bandwidth fiber optic channel that provides the high-speed connectivity required for data sharing between a coupling facility and the central processor complexes directly attached to it.

coupling services

In a sysplex, the functions of XCF that transfer data and status between members of a group residing on one or more MVS systems in the sysplex.

CP

Central processor.

CPC

Central processor complex.

cross-system coupling facility (XCF)

XCF is a component of MVS that provides functions to support cooperation between authorized programs running within a sysplex.

cross-system extended services (XES)

XES is a set of services that allow authorized applications or subsystems running in a sysplex to share data using a coupling facility.

cross-system restart

If a system fails, automatic restart management restarts elements on another eligible system in the sysplex.

CTC

Channel-to-channel.

D

DASD

Direct access storage device.

data sharing

The ability of concurrent subsystems (such as DB2 or IMS DB) or application programs to directly access and change the same data while maintaining data integrity.

DB2

DATABASE2.

DFSMS/MVS

Data Facility Storage Management Subsystem.

E**element**

A batch job or started task that uses the IXCARM macro to register for automatic restart management services. In the event of an unexpected failure of the element itself or of the system it is running on, MVS automatically restarts the element.

Enterprise Systems Connection (ESCON)

A set of products and services that provides a dynamically connected environment using optical cables as a transmission medium.

ESCD

ESCON Director.

ESCON

Enterprise Systems Connection.

ETR

External Time Reference. See also *Sysplex Timer*.

G**global resource serialization**

A function that provides an MVS serialization mechanism for resources (typically data sets) across multiple MVS images.

global resource serialization complex

One or more MVS systems that use global resource serialization to serialize access to shared resources (such as data sets on shared DASD volumes).

I**IMS DB**

Information Management System Database Manager.

J**JES2**

Job Entry Subsystem 2.

JES3

Job Entry Subsystem 3.

L**LIC**

Licensed Internal Code.

logical partition (LP)

A subset of the processor hardware that is defined to support an operating system. See also *logically partitioned (LPAR) mode*.

logically partitioned (LPAR) mode

A central processor complex (CPC) power-on reset mode that enables use of the PR/SM feature and allows an operator to allocate CPC hardware resources (including central processors, central storage, expanded storage, and channel paths) among logical partitions.

LP

Logical partition.

LPAR

Logically partitioned (mode).

M**member**

A specific function (one or more modules/routines) of a multisystem application that is defined to XCF and assigned to a group by the multisystem application. A member resides on one system in the sysplex and can use XCF services to communicate (send and receive data) with other members of the same group.

mixed complex

A global resource serialization complex in which one or more of the systems in the global resource serialization complex are not part of a multisystem sysplex.

monoplex

A sysplex consisting of one system that uses a sysplex couple data set.

multiprocessing

The simultaneous execution of two or more computer programs or sequences of instructions. See also *parallel processing*.

multiprocessor (MP)

A CPC that can be physically partitioned to form two operating processor complexes.

multisystem application

An application program that has various functions distributed across MVS images in a multisystem environment.

multisystem environment

An environment in which two or more MVS images reside in one or more processors, and programs on one image can communicate with programs on the other images.

multisystem sysplex

A sysplex in which two or more MVS images are allowed to be initialized as part of the sysplex. See also *single-system sysplex*.

MVS image

A single occurrence of the MVS/ESA operating system that has the ability to process work.

MVS system

An MVS image together with its associated hardware, which collectively are often referred to simply as a system, or MVS system.

MVS/ESA

Multiple Virtual Storage/ESA.

N**n-way**

The number (*n*) of CPs in a CPC. For example, a 6-way CPC contains six CPs.

P**parallel processing**

The simultaneous processing of units of work by many servers. The units of work can be either transactions or subdivisions of large units of work (batch).

PR/SM

Processor Resource/Systems Manager.

Processor Resource/Systems Manager (PR/SM)

The feature that allows the processor to use several MVS images simultaneously and provides logical partitioning capability. See also *LPAR*.

persistent command text

The command text that MVS saves when a started task is started, and uses to restart the job through automatic restart management.

persistent JCL

The JCL that MVS saves when a job is started, and uses to restart the job through automatic restart management.

R**RACF**

Resource Access Control Facility.

register

To issue the IXCARM macro to become an element of automatic restart management, an MVS recovery function that restarts jobs in the event of an unexpected failure.

restart group

A group of related jobs, registered as elements of automatic restart management, that need to be restarted together on the same system, should a system fail unexpectedly.

RMF

Resource Measurement Facility.

S**SFM**

Sysplex failure management

single-system sysplex

A sysplex in which only one MVS system is allowed to be initialized as part of the sysplex. In a single-system sysplex, XCF provides XCF services on the system but does not provide signaling services between MVS systems. See also *multisystem sysplex*, *XCF-local mode*.

SMS

Storage Management Subsystem.

structure

A construct used by MVS to map and manage storage on a coupling facility.

sysplex

A set of MVS systems communicating and cooperating with each other through certain multisystem hardware components and software services to process customer workloads. See also *MVS system*.

sysplex couple data set

A couple data set that contains sysplex-wide data about systems, groups, and members that use XCF services. All MVS systems in a sysplex must have connectivity to the sysplex couple data set. See also *couple data set*.

Sysplex Timer

An IBM unit that synchronizes the time-of-day (TOD) clocks in multiple processors or processor sides. External Time Reference (ETR) is the MVS generic name for the IBM Sysplex Timer (9037).

System/390® microprocessor cluster

A configuration that consists of central processor complexes (CPCs) and may have one or more coupling facilities.

W**WLM**

MVS workload manager.

X**XCF**

Cross-system coupling facility.

XCF PR/SM policy

In a multisystem sysplex on PR/SM, the actions that XCF takes when one MVS system in the sysplex fails. This policy provides high availability for multisystem applications in the sysplex.

XCF-local mode

The state of a system in which XCF provides limited services on one system and does not provide signaling services between MVS systems. See also *single-system sysplex*.

XES

Cross-system extended services

Index

A

- accessibility
 - contact IBM [469](#)
- ACOUPLE keyword [35](#)
- administrative data utility
 - coding [336](#)
 - description [335](#)
 - return code [397](#)
 - using [335](#)
 - utility control statements [336](#)
- ALLOWAUTOALT keyword [351](#)
- ALLOWREALLOCATE keyword [357](#)
- APPC/MVS
 - finding information [242](#)
- assistive technologies [469](#)
- authorization requirements
 - system logger [273](#)
- AUTODELETE parameter
 - system logger [272](#), [273](#)
- automatic deletion policy
 - system logger
 - with a resource manager [273](#)
- automatic restart management
 - checkpoint/restart [193](#)
 - parameters [318](#)
 - planning [178](#)
 - requirements for [193](#)
 - setting up a policy [193](#)
- automatic structure alter [63](#)
- automatically switchable tape devices
 - benefits of [467](#)
 - definition [467](#)
- average log block size
 - system logger [281](#)
- AVGBUFSIZE
 - system logger [281](#)

C

- capacity planning
 - signaling path [162](#)
- CFRM
 - system logger
 - coupling facility attributes in CFRM [290](#)
 - define the coupling facility structure attributes in CFRM [289](#)
- CFRM (coupling facility resource management)
 - couple data set, formatting [37](#)
 - policies [67](#)
- CICS log manager
 - finding information [239](#)
- CLASS keyword [108](#), [116](#)
- class length, tuning [151](#)
- CLASSDEF statement [108](#), [109](#), [111](#), [152](#)
- CLASSLEN keyword [111](#), [152](#)
- cleanup interval [179](#)

- CLEANUP keyword [179](#)
- CLOCKxx parmlib member
 - planning [14](#), [23](#)
- component trace [19](#)
- configurations
 - multisystem
 - with existing complex [204](#)
 - without existing complex [203](#)
 - on PR/SM, sysplex on LPARs [423](#)
 - single system [202](#)
 - under VM [419](#)
- connector persistence
 - description [425](#)
- considerations
 - encrypting log stream data sets [236](#)
- CONSOLxx parmlib member
 - planning [22](#)
- contact
 - z/OS [469](#)
- control interval size [282](#)
- couple data set
 - ARM, formatting [331](#)
 - CFRM parameters [319](#)
 - CFRM, formatting [331](#)
 - connectivity requirements [38](#)
 - considerations
 - automatic restart management [27](#), [37](#)
 - CFRM [27](#), [37](#)
 - LOGR [27](#), [37](#)
 - security [31](#)
 - SFM [27](#), [37](#)
 - sysplex [27](#)
 - system joining sysplex [35](#)
 - WLM [27](#), [37](#)
 - format [313](#)
 - joining a sysplex [35](#)
 - LOGR [218](#)
 - LOGR parameters [320](#)
 - LOGR, formatting [327](#), [331](#)
 - performance considerations [30](#)
 - planning [27](#)
 - policies
 - automatic restart management [5](#)
 - CFRM [5](#)
 - LOGR [5](#)
 - SFM [5](#), [27](#)
 - WLM [5](#), [27](#)
 - SFM parameters [328](#)
 - SFM, formatting [332](#)
 - SMS-managed volume [31](#)
 - switching [28](#)
 - system logger [218](#)
 - WLM parameters [329](#)
 - WLM, formatting [332](#)
 - z/OS UNIX parameters [328](#)
 - z/OS UNIX, formatting [332](#)
- COUPLE statement [9](#), [35](#), [105](#), [179](#), [181](#)

- COUPLE system parameter
 - options [17](#)
- COUPLExx parmlib member
 - CLEANUP keyword [179](#)
 - component trace options [19](#)
 - COUPLE statement [179](#)
 - examples
 - defaults, using [117](#)
 - signaling paths, defining through coupling facility [123](#)
 - signaling through coupling facility and CTC connections [125](#)
 - transport class, creating for group [121](#)
 - transport class, creating for large messages [119](#)
 - failure-related information [177](#)
 - INTERVAL keyword [178](#)
 - modifying values [22](#)
 - OPNOTIFY keyword [179](#)
 - status update missing condition [181](#)
 - syntax [18](#)
 - with PLEXCFG=MONOPLEX [10](#)
 - with PLEXCFG=MULTISYSTEM [9](#)
 - with PLEXCFG=XCFLOCAL [9](#)
- coupling facility
 - adding [429](#)
 - control code support [42](#)
 - description [41](#), [49](#)
 - dump space [65](#)
 - failure-independence [47](#)
 - guidelines [425](#)
 - identifying [49](#)
 - non-volatility [47](#)
 - operator commands [89](#)
 - planning, physical [42](#)
 - POPULATECF function [90](#)
 - rebuilding structures [439](#)
 - reconfiguring [425](#)
 - replacing [425](#)
 - resetting [88](#)
 - storage allocation [87](#)
 - storage increment [88](#)
 - structure threshold for rebuild [59](#)
 - system logger
 - damage [305](#)
 - failure [305](#), [308](#)
 - full condition [308](#)
 - loss of connectivity to [305](#)
 - system logger data on [214](#)
 - types of storage [87](#)
- coupling facility structure
 - identifying [49](#)
 - space needed, per log stream
 - calculating [255](#)
 - formula [255](#)
 - system logger
 - space needed, per log stream [255](#)

D

- DASD log data set
 - space [309](#)
- DASD log data sets
 - increasing space for [262](#)
 - SMS management of [221](#)

- DASD-only log stream
 - DASDONLY [281](#)
 - system logger [281](#)
 - system logger applications
 - staging data sets [265](#)
 - system logger data on [215](#)
- DASDONLY
 - LOGR policy [281](#)
 - system logger [281](#)
- DATA statement [336](#)
- dedicated tape device
 - definition [467](#)
- DEFAULT statement [33](#)
- DEFINE POLICY statement [338](#)
- deletion utility for XCF group members
 - description [401](#)
- DEVICE keyword [102](#)
- DFSMS components and subsystems
 - group names and members
 - DFSMS [33](#)
- DFW (DASD fast write) feature
 - using [30](#)
- dump space
 - in coupling facility [65](#)
- duplexing
 - conditional [284](#)
 - system logger
 - example [285](#)
 - methods [283](#)
 - unconditional [284](#)
- duplexing rebuild
 - common problems [78–80](#)
 - controlling [73](#)
 - migration [71](#)
 - monitoring [74](#)
 - planning [69](#)
 - requirements [69](#)
 - system-managed [74](#)
- duplexing rebuild process
 - policy changes [73](#)
- dynamic IO coupling facility changes [447](#)

E

- ENFORCEORDER keyword [356](#)
- entry-to-element ratio [220](#)
- ETRMODE statement [15](#)
- ETRZONE statement [15](#)
- example
 - log stream configuration [250](#)
 - merging log data across the sysplex [250](#)
 - system logger
 - duplexing [285](#)
 - system logger applications
 - multiple log streams in the same coupling facility
 - structure [251](#)
- exclusion list
 - in CFRM policy [56](#)

F

- failed-persistent connections
 - deleting [447](#)

failure detection interval [178](#)

format utility

examples of using [331](#)

XCF couple data set

coding [314](#)

description [313](#)

return code [330](#)

sample output [332](#)

using [313](#)

formatting

LOGRY or LOGRZ couple data set [328](#)

single system scope [328](#)

FULLTHRESHOLD keyword [351](#)

G

global resource serialization

installation [204](#)

planning [11](#)

GROUP keyword [109](#)

GRS system parameter

planning [11](#)

GRSCNF system parameter

planning [13](#), [18](#)

GRSRNL system parameter

planning [13](#)

H

HYPERPAV storage subsystems

planning to use [24](#)

I

IEASYSxx system parameter

planning [9](#), [23](#)

IECIIOSxx parmlib member

planning [24](#)

IMS log manager

finding information [242](#)

INTERVAL keyword [178](#), [189](#)

IXCDELET utility

output message [401](#)

IXCDELET utility program [401](#)

IXCL1DSU utility program [313](#)

IXCMIAPU utility program

examples of using

ARM policy data [397](#)

CFRM policy data [398](#)

SFM policy data [399](#)

IXGINVNT service

duplexing

DUPLEXMODE parameter [284](#)

STG_DUPLEX parameter [284](#)

DUPLEXMODE parameter [284](#)

STG_DUPLEX parameter [284](#)

J

JCL (job control language)

z/OS couple data set format utility
[331](#)

JCL control statements [335](#)

JES3

complex in a sysplex [34](#)

K

keyboard

navigation [469](#)

PF keys [469](#)

shortcut keys [469](#)

KEYRNOTIFYDELAY keyword [356](#)

L

LISTNOTIFYDELAY keyword [355](#)

LOCAL sysplex name [17](#)

local time, adjusting [14](#)

LOCALMSG statement [111](#)

log data

deleting [291](#)

on DASD log data sets [214](#), [215](#)

log data deletion

offloading [226](#)

log data sets

allocation [215](#)

data class [267](#)

define DASD space for [261](#)

deleting [291](#)

GRSRNL inclusion list [267](#)

naming convention for [258](#)

share options [267](#)

SMS environment for [266](#)

log stream

calculating

space needed, per log stream [255](#)

configuration, examples [250](#)

merging log data across the sysplex

example [250](#)

multiple log streams in the same coupling facility

structure

example [251](#)

resource manager managed [289](#)

upgrading

from DASD-only to coupling facility [295](#)

log stream subsystem

start [291](#)

LOGR

configuration resource [231](#)

couple data set [218](#)

policy [218](#)

resource configuration [246](#)

LOGR couple data set

define the coupling facility structure attributes in CFRM
[289](#)

DSEXTENT parameter [262](#)

format and define the LOGR couple data sets [276](#)

prevent LOGR CDS access on a z/OS image [277](#)

LOGR policy

DASDONLY [281](#)

define coupling facility structures in [280](#)

define log streams in [280](#)

defining a log stream [359](#)

defining a structure [371](#)

deleting a log stream [384](#)

- LOGR policy (*continued*)
 - deleting a structure [384](#)
 - listing a log stream
 - continue [386](#)
 - wildcards [385](#)
 - listing a structure
 - continue [386](#)
 - wildcards [385](#)
 - updating a log stream [371](#)
- LOGR subsystem
 - read log data in data set format [291](#)
 - start [291](#)
- logrec log stream
 - finding information [239](#), [241](#)
- LOGRY and LOGRZ couple data sets
 - policies [219](#)
 - single-system scope [219](#)
- LOGRY or LOGRZ
 - couple data set versioning [327](#)
 - for format utility [326](#)
 - new format level [327](#)
 - single-system scope parameters [326](#)
- LPAR (logical partition) configuration [424](#)

M

- managing log data [272](#)
- MAXMSG keyword [112](#), [116](#)
- MAXSYSTEM keyword [210](#)
- message buffer
 - space [112](#)
 - tuning [151](#)
- message traffic, partitioning [153](#)
- message-based processing [68](#)
- MGTCCLAS keyword [32](#)
- migration
 - system-managed asynchronous duplexing [72](#)
- MINSIZE keyword [349](#)
- MODIFY keyword [22](#)
- modifying channel path access [447](#)
- multisystem sysplex
 - description [204](#), [208–210](#)
 - greater than eight systems [210](#)
 - installation [203](#), [204](#), [208–210](#)
 - on one processor [208](#)
 - on PR/SM [209](#)
 - with coupling facility [209](#)
- MVS components and subsystems
 - group names and members
 - APPC [33](#)
 - console services [33](#)
 - DAE [33](#)
 - JES2 MAS [34](#)
 - RRS [34](#)
 - VLF [34](#)
 - WLM [34](#)
 - XCF [34](#)
 - XES [34](#)

N

- navigation
 - keyboard [469](#)

O

- offloading
 - system logger data [222](#), [225](#)
- operator notification interval [178](#)
- OPERLOG log stream
 - finding information [239](#), [240](#)
- OPNOTIFY keyword [179](#)

P

- parmlib members
 - CLOCKxx [208](#)
 - CONSOLxx [33](#), [316](#)
 - COUPLExx [4](#), [9](#), [35](#), [152](#), [177](#), [201](#)
 - GRSCNFxx [204](#)
 - GRSRNL [14](#)
 - GRSRNLxx [204](#)
 - IEASYMxx [9](#)
 - IEASYSxx [4](#), [9](#)
 - XCFPOLxx [177](#), [188](#)
- PATHIN statement [9](#), [105](#), [112](#), [115](#)
- PATHOUT statement [9](#), [105](#), [111](#), [112](#)
- PCOUPLE keyword [35](#)
- performance
 - couple data set considerations [30](#)
- persistence
 - description [425](#)
- planning
 - availability [1](#), [177](#)
 - cleanup interval [179](#)
 - couple data sets [27](#)
 - failure detection interval [178](#)
 - IEASYSxx parmlib member [9](#), [23](#)
 - inbound signaling path [102](#)
 - message buffer space [112](#)
 - operator notification interval [178](#)
 - outbound signaling path [102](#)
 - PR/SM reconfigurations [186](#)
 - recovery [177](#)
 - signaling services [99](#)
 - status update missing condition [181](#)
 - system logger applications
 - automatically allocated staging data sets [263](#)
 - configuration [247](#)
 - coupling facility space for log streams [255](#)
 - DASD space [260](#)
 - DASD space for log data sets [261](#)
 - DASD space for staging data sets [263](#)
 - data class [267](#)
 - GRSRNL inclusion list [267](#)
 - map log streams to structures [248](#)
 - naming convention for system logger resources [258](#)
 - naming conventions for coupling facility structures [260](#)
 - naming conventions for log stream and staging data sets [258](#)
 - number of log streams [247](#)
 - share options [267](#)
 - SMS environment for DASD data sets [266](#)
 - transport class [106](#)
 - XCFPOLxx, controlling availability and recovery [187](#)
- PLEXCFG system parameter
 - planning [9](#)

- policy
 - automatic restart management [178](#)
 - CFRM [49](#)
 - defining and activating [37](#)
 - LOGR [218](#)
 - SFM [177](#), [180](#)
 - system logger [218](#)
 - updating [38](#)
- POLICY control statement [335](#)
- policy-based processing [68](#)
- POPULATECF function [90](#)
- PR/SM (Processor Resource/Systems Manager)
 - changing the XCF policy [188](#)
 - in sysplex configuration [423](#)
 - sysplex migration [209](#)
- preference list
 - in CFRM policy [56](#)
- preparing for
 - system logger applications
 - automatically allocated staging data sets [263](#)
 - configuration [247](#)
 - coupling facility space for log streams [255](#)
 - DASD space [260](#)
 - DASD space for log data sets [261](#)
 - DASD space for staging data sets [263](#)
 - data class [267](#)
 - GRSRNL inclusion list [267](#)
 - map log streams to structures [248](#)
 - naming convention for system logger resources [258](#)
 - naming conventions for coupling facility structures [260](#)
 - naming conventions for log stream and staging data sets [258](#)
 - number of log streams [247](#)
 - share options [267](#)
 - SMS environment for DASD data sets [266](#)
- PRSMPOLICY keyword [188](#)

R

- read
 - log data in data set format
 - LOGR subsystem [291](#)
- REBUILDPERCENT keyword [59](#), [352](#)
- recovery
 - system logger
 - coupling facility failure [305](#)
 - coupling facility full [308](#)
 - DASD log data set space fills [309](#)
 - example, system failure [252](#), [253](#)
 - log stream damage [309](#)
 - peer connector [251](#), [301](#)
 - staging data set full [308](#)
 - sysplex failure [301](#)
 - system failure [251](#), [301](#)
 - system failure, single system sysplex [301](#)
 - system logger address space failure [302](#)
 - unrecoverable I/O errors [310](#)
- rejoining a sysplex [212](#)
- removing a system from a sysplex [210](#)
- requirements
 - system logger [245](#), [246](#)
 - system logger coupling facility [245](#)

- resource configuration
 - LOGR [246](#)
 - system logger [246](#)
- resource manager
 - associating a log stream with [289](#)
 - system logger
 - with automatic deletion [273](#)
- retention period
 - system logger [262](#), [272](#)
- RETPD parameter
 - system logger [272](#)
- RETRY keyword [105](#), [116](#)
- RMAX keyword [33](#)
- RMF (Resource Measurement Facility)
 - XCF Activity Report [164](#)
- RRS logger application
 - finding information [243](#)

S

- SETXCF command [22](#)
- SFM (sysplex failure management) [27](#)
- shortcut keys [469](#)
- signaling
 - description [6](#)
 - examples
 - creating class for large messages [119](#)
 - defaults, using [117](#)
 - signaling paths, defining through coupling facility [123](#)
 - signaling through coupling facility and CTC connections [125](#)
 - transport class, creating for group [121](#)
 - options [100](#)
 - recovery capabilities [102](#)
 - services
 - CFRM policy [102](#)
 - COUPLExx parameters [105](#)
 - list structure, determining size [104](#)
 - message buffer space [112](#)
 - performance [149](#)
 - setting up [102](#)
 - signaling paths [102](#)
 - single point of failure, avoiding [103](#)
 - sizes, calculating [407](#)
 - transport classes [106](#)
 - tuning [149](#), [150](#)
 - XCF groups [106](#)
 - systems management [102](#)
- signaling path
 - capacity planning [158](#), [162](#)
 - CLASS keyword [111](#)
 - defining through coupling facility [123](#)
 - inbound [102](#)
 - message counts [159](#)
 - none in class [160](#)
 - outbound [102](#)
 - performance [159](#)
 - planning [102](#)
 - planning for transport class [111](#)
 - restarts [160](#)
 - retry limit [105](#)
 - STRNAME keyword [112](#)
 - tuning [158](#)

- signalling path
 - IPL
 - problem [128](#)
 - recovery [127](#)
- SIMETRID statement [15](#)
- single point of failure
 - system logger [283](#)
- SITE keyword [348](#)
- SMF logger application
 - finding information [243](#)
- SMS (Storage Management Subsystem) [31](#)
- SMS parameters
 - system logger
 - ACS routines, interaction [282](#)
 - LIKE parameter, interaction [282](#)
- staging data sets
 - automatically allocated [263](#)
 - connectivity to [263](#), [265](#)
 - DASD-only log stream [265](#)
 - data class [267](#)
 - define DASD space for [263](#)
 - full condition [308](#)
 - GRSRNL inclusion list [267](#)
 - naming convention for [258](#)
 - share options [267](#)
 - size [226](#)
 - SMS environment for [266](#)
 - system logger [221](#)
- START keyword [115](#)
- status update missing
 - planning for [181](#)
- STOP keyword [115](#)
- STORCLAS keyword [32](#)
- STRNAME keyword [112](#)
- structure
 - deleting persistent structures [447](#)
 - rebuilding in a coupling facility [439](#)
- Structure Full Monitoring [60](#)
- structure persistence
 - description [425](#)
- structure rebuild
 - initiated by MVS [59](#)
 - process [439](#)
- summary of changes [xxi](#)
- sysplex
 - automatic restart management [193](#)
 - configurations [419](#)
 - couple data set requirement [1](#)
 - COUPLExx defaults, using [117](#)
 - description [1](#)
 - format utility [313](#)
 - hardware requirements [1](#)
 - illustration [2](#)
 - installing [7](#)
 - local time, adjusting [14](#)
 - MVS systems, installing [201](#)
 - parameter values (Parmlib) [4](#)
 - parmlib member, required values [9](#)
 - PDSEs, sharing [33](#)
 - policies
 - defining in couple data set [5](#)
 - description [5](#)
 - purpose [5](#)
 - SFM [177](#)

- sysplex (*continued*)
 - rejoining [212](#)
 - services, planning [99](#)
 - SFM policy
 - defining [187](#)
 - PR/SM reconfigurations [180](#)
 - signaling connectivity failures [180](#), [183](#)
 - system availability and recovery, controlling [180](#)
 - system failure responses [180](#)
 - signaling
 - description [6](#)
 - software requirements [1](#)
 - Sysplex Timer [1](#)
 - tuning [7](#), [149](#)
 - XCF [1](#)
- Sysplex Timer
 - description [2](#)
 - logical time offset [7](#)
- SYSPRINT DD statement [314](#)
- system logger
 - APPC/MVS
 - finding information [242](#)
 - authorization requirements [273](#)
 - AUTODELETE parameter [272](#), [273](#)
 - automatic deletion policy [262](#), [272](#), [273](#)
 - automatic deletion with a resource manager [273](#)
 - average log block size [281](#)
 - AVGBUFSIZE [281](#)
 - CICS log manager
 - finding information [239](#)
 - component [217](#)
 - configuration
 - example [250](#), [251](#)
 - illustration [215](#), [216](#)
 - consistency requirements [246](#)
 - couple data set [218](#)
 - coupling facility requirements [245](#)
 - coupling facility size [226](#), [255](#)
 - DASD log data sets
 - increasing space for [262](#)
 - DASD-only log stream [215](#), [281](#)
 - DASDONLY [281](#)
 - definitions in the LOGR policy [280](#)
 - deleting log data [291](#)
 - deleting log data sets [291](#)
 - duplexing
 - DASD-only [222](#)
 - example [285](#)
 - methods [283](#)
 - duplexing log data
 - DASD-only [222](#)
 - from a coupling facility log stream [221](#)
 - encrypting log stream data sets [236](#)
 - entry-to-element ratio [220](#)
 - example
 - duplexing [285](#)
 - examples [250](#)
 - formatting a single-system scope LOGRY or LOGRZ couple data set [328](#)**
 - IMS log manager
 - finding information [242](#)
 - interim storage [253](#)
 - log data deletion [226](#)
 - log data on the coupling facility [214](#)

system logger (*continued*)

- log data sets
 - allocation [215](#)
 - SMS management of [221](#)
- log stream
 - resource manager managed [289](#)
 - upgrading [295](#)
- LOGR couple data set
 - DSEXTENT parameter [262](#)
- LOGR policy [218](#), [219](#)
- LOGR resource configuration [246](#)
- LOGR subsystem [291](#)
- logrec log stream
 - finding information [241](#)
- LOGRY and LOGRZ couple data sets policies** [219](#)
- LOGRY or LOGRZ couple data set versioning - new format levels** [327](#)
- LOGRY or LOGRZ couple data sets for a single-system scope within a sysplex** [279](#)
- LOGRY or LOGRZ single-system scope parameters for format utility** [326](#)
- low offload threshold [223](#)
- low threshold [223](#)
- managing log data [272](#)
- offloading
 - coupling facility size [226](#), [255](#)
 - log data [222](#)
 - log data deletion [226](#)
 - staging data set size [226](#)
 - thresholds [225](#)
- OPERLOG log stream
 - finding information [240](#)
- peer connector
 - in recovery [251](#), [301](#)
- planning
 - automatically allocated staging data sets [263](#)
 - configuration [247](#)
 - coupling facility space for log streams [255](#)
 - DASD space for log data sets [261](#)
 - DASD space for staging data sets [263](#)
 - data class [267](#)
 - GRSRNL inclusion list [267](#)
 - map log streams to structures [248](#)
 - naming convention for system logger resources [258](#)
 - naming conventions for coupling facility structures [260](#)
 - naming conventions for log stream and staging data sets [258](#)
 - number of log streams [247](#)
 - share options [267](#)
 - SMS environment for DASD data sets [266](#)
- policy [218](#)
- preparing for [243](#)
- read
 - log data in data set format [291](#)
- recovery
 - coupling facility failure [305](#)
 - coupling facility full [308](#)
 - DASD log data set space fills [309](#)
 - example, system failure [252](#), [253](#)
 - log stream damage [309](#)
 - staging data set full [308](#)
 - sysplex failure [301](#)
 - system failure [251](#), [301](#)

system logger (*continued*)

- recovery (*continued*)
 - system failure, single system sysplex [301](#)
 - system logger address space failure [302](#)
 - unrecoverable I/O errors [310](#)
- requirements [245](#)
- resource configuration [246](#)
- retention period [262](#), [272](#)
- RETPD parameter [272](#)
- RRS application
 - finding information [243](#)
- single point of failure [283](#)
- SMF application
 - finding information [243](#)
- SMS parameters and ACS routines [282](#)
- SMS parameters and LIKE parameter [282](#)
- SMS requirements [246](#)
- staging data sets
 - connectivity to [263](#), [265](#)
 - size [226](#)
- system logger applications [213](#)
- thresholds [228](#)
- thresholds and offloading [225](#)
- system logger address space
 - authorization requirements [273](#)
- system logger applications
 - authorization requirements [273](#)
 - DASD-only log stream
 - staging data sets [265](#)
 - DUPLEXMODE parameter [284](#)
 - finding information [239](#)
 - planning [243](#)
 - preparing for [243](#)
 - preparing to use
 - automatically allocated staging data sets [263](#)
 - configuration [247](#)
 - coupling facility space for log streams [255](#)
 - DASD space for log data sets [261](#)
 - DASD space for staging data sets [263](#)
 - data class [267](#)
 - map log streams to structures [248](#)
 - naming convention for system logger resources [258](#)
 - naming conventions for coupling facility structures [260](#)
 - naming conventions for log stream and staging data sets [258](#)
 - number of log streams [247](#)
 - share options [267](#)
 - SMS environment for DASD data sets [266](#)
 - STG_DUPLEX parameter [284](#)
- system parameters for IEASYSxx
 - CLOCK [14](#)
 - COUPLE [17](#)
 - GRS [11](#)
 - GRSCNF [13](#)
 - GRSRNL [13](#)
 - IXGCNF [18](#)
 - PLEXCFG [9](#)

T

- tape device
 - in a sysplex [467](#)
- time stamps, synchronizing [14](#)

TIMEZONE statement [14](#)

TOD clock

resetting, consequences [14](#)

setting backward [17](#)

setting forward [17](#)

trademarks [474](#)

transport class

assigning path [111](#)

CLASSLEN keyword [111](#)

creating class for group [121](#)

creating class for large messages [119](#)

default [109](#)

defining [108](#)

GROUP keyword [109](#)

message length [111](#)

planning [106](#)

tuning [160](#)

undesignated groups [109](#)

XCF groups, assigning [109](#)

tuning sysplex

capacity planning [162](#)

class length [151](#)

introduction [149](#)

message buffer size [151](#)

message buffer space

adjusting [155](#)

description [154](#)

inbound, analyzing lack [157](#)

outbound, analyzing lack [157](#)

RMF reports, using to analyze [156](#)

signaling path

capacity planning [158](#), [162](#)

message counts [159](#)

none in class [160](#)

performance [159](#)

restarts [160](#)

signaling service [149](#), [150](#)

transport classes [160](#)

U

upgrading

log stream

from DASD-only to coupling facility [295](#)

user interface

ISPF [469](#)

TSO/E [469](#)

Using LOGRY or LOGRZ couple data sets

single system scope within a sysplex [279](#)

using couple data sets for a single system scope within a
sysplex [279](#)

utility program

to create policy data [335](#)

to delete XCF group members [401](#)

to format couple data set [313](#)

V

value ranges/limits

LOGR [231](#)

system logger [231](#)

VM System Product

sysplex migration [208](#)

VOLSER keyword [32](#)

W

WLM (workload management) [27](#)

X

XCF (cross-system coupling facility)

component [1](#)

configurations [419](#)

format utility [313](#)

groups

assigning to transport class [109](#)

description [106](#)

GROUP keyword [109](#)

message characteristics [106](#)

names [106](#)

undesignated [109](#)

local mode [9](#)

PDSEs, sharing [33](#)

PR/SM policy

changing [188](#)

one processor, example [188](#), [191](#)

PRSMPOLICY keyword [188](#)

two processors, example [189](#)

RETAIN option [211](#)

SETXCF command

add path [115](#)

changing the PR/SM policy
[188](#)

delete path [115](#)

START keyword [115](#)

STOP keyword [115](#)

tuning a sysplex [149](#)

XCF-local mode [201](#)

XCFPOLxx parmlib member

planning [23](#)



Product Number: 5655-ZOS

SA23-1399-70

