

z/OS
3.2

*MVS Interactive Problem Control System
(IPCS) User's Guide*



Note

Before using this information and the product it supports, read the information in [“Notices” on page 153](#).

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..... | vii |
| Tables..... | xi |
| About this information..... | xiii |
| Who should use this information..... | xiii |
| z/OS information..... | xiii |
| How to provide feedback to IBM..... | xv |
| Summary of changes..... | xvii |
| Summary of changes for z/OS 3.2..... | xvii |
| Summary of changes for z/OS 3.1..... | xvii |
| Part 1. Getting started with IPCS..... | 1 |
| Chapter 1. Introduction to IPCS..... | 3 |
| What You Can Do with IPCS..... | 3 |
| Analyzing Dumps..... | 3 |
| Analyzing Traces..... | 3 |
| IPCS Concepts..... | 4 |
| Processing Dumps, Traces, and Other Sources..... | 4 |
| Using IPCS at the Terminal..... | 6 |
| Saving Dump Analysis Results..... | 7 |
| Providing Format and Analysis Support..... | 8 |
| Other Pieces of IPCS..... | 8 |
| How IPCS Works as a TSO/E Application..... | 9 |
| IPCS Command Processing..... | 9 |
| Running Different Levels of IPCS..... | 10 |
| IPCS Messages and User Completion Codes..... | 11 |
| Creating IPCS Analysis and Formatting Routines..... | 11 |
| Chapter 2. Starting IPCS..... | 13 |
| Starting IPCS..... | 13 |
| Customized Access..... | 13 |
| Starting IPCS with Customized Access..... | 13 |
| Starting IPCS without Customized Access..... | 14 |
| Setting Session Defaults..... | 15 |
| IPCS Default Values..... | 15 |
| Specifying the Source..... | 17 |
| Initializing Dumps and Traces..... | 17 |
| Dump Data Set Initialization..... | 17 |
| Trace Data Set Initialization..... | 18 |
| Requesting IPCS Functions..... | 18 |
| Part 2. Using IPCS functions..... | 21 |
| Chapter 3. Using the IPCS Dialog..... | 23 |
| IPCS Primary Option Menu..... | 23 |

| | |
|---|-----|
| Selecting an Option..... | 24 |
| Option 0 — DEFAULTS..... | 24 |
| Option 1 — BROWSE..... | 26 |
| Option 2 — ANALYSIS..... | 26 |
| Option 3 — UTILITY..... | 31 |
| Option 4 — INVENTORY..... | 38 |
| Option 5 — SUBMIT..... | 40 |
| Option 6 — COMMAND..... | 41 |
| Option T — TUTORIAL..... | 42 |
| IPCS Dialog Panels..... | 43 |
| Selection and Data Entry Panels..... | 43 |
| Pointer and Storage Panels..... | 43 |
| Dump Display Reporter Panel..... | 44 |
| Browsing a Dump Data Set..... | 46 |
| Entry Panel..... | 46 |
| Pointer Panel..... | 47 |
| Storage Panel..... | 50 |
| Examining 64-bit Dump Data Sets..... | 56 |
| Getting Online Help in the IPCS Dialog..... | 59 |
| Help Using IPCS Panels and Primary and Line Commands..... | 59 |
| Help Entering IPCS Subcommands..... | 59 |
| Help Reading Formatted Dump Reports..... | 60 |
| Help Choosing a Component Analysis Option..... | 60 |
| ISPF Operations on the IPCS Dialog..... | 60 |
| Splitting the Display Screen..... | 60 |
| Entering Primary and Line Commands..... | 60 |
| Using Program Function (PF) Keys..... | 60 |
| Customizing the IPCS Dialog..... | 60 |
| Customizing the Primary Command Definitions..... | 61 |
| Customizing Your Program Function (PF) Keys..... | 61 |
| Using Data Privacy for Diagnostics..... | 62 |
| Requesting an ANALYZE run..... | 67 |
| Requesting a REPORT run..... | 74 |
| Requesting a FEEDBACK run..... | 74 |
| Requesting an INGEST run..... | 75 |
| Requesting an EXTRACT run..... | 80 |
| Chapter 4. Using the User and Sysplex Dump Directories..... | 83 |
| Using a User Dump Directory..... | 83 |
| Using the Sysplex Dump Directory..... | 83 |
| Managing Dump Directories..... | 84 |
| Preparing a Dump Directory..... | 84 |
| Maintaining a Dump Directory..... | 87 |
| Changing the Characteristics of a Dump Directory..... | 89 |
| Using the Symbol Table..... | 90 |
| Using the Storage Map..... | 94 |
| Changing Your Current User Dump Directory..... | 96 |
| Administering the Sysplex Dump Directory..... | 97 |
| Chapter 5. Describing Storage in a Dump..... | 99 |
| Categories of Data Description Parameters..... | 99 |
| Specifying Data Description Parameters..... | 101 |
| Resolving Default Data Description Parameters..... | 101 |
| Using Indirect Addressing with Symbols..... | 102 |
| Chapter 6. Using IPCS REXX EXECs and CLISTs..... | 103 |
| Invoking REXX EXECs and CLISTs from an IPCS Session..... | 103 |
| Using ALTLIB..... | 104 |

| | |
|--|-----|
| Starting an IPCS Session from a REXX EXEC or CLIST..... | 105 |
| Using IPCS-Supplied CLISTs to Analyze Dumps..... | 105 |
| CLISTs that Perform Initial Dump Analysis..... | 106 |
| CLIST that Runs the Save Area Chain..... | 107 |
| Using IPCS-Supplied CLISTs to Print Dump Analysis Reports..... | 107 |
| CLISTs that Print Screening Dump Reports..... | 107 |
| CLISTs that Print Detailed Dump Reports..... | 107 |
| Using an IPCS-Supplied CLIST to Print Dump Storage..... | 107 |
| IPCS-Supplied Sample REXX EXECs and CLISTs..... | 108 |
| Retrieving Data from a Dump..... | 108 |
| Processing Control Blocks..... | 108 |
| Printing Dump Storage..... | 108 |
| Writing IPCS REXX EXECs..... | 109 |
| ADDRESS IPCS Instruction..... | 109 |
| Putting the Pieces Together..... | 114 |
| Debugging IPCS REXX EXECs..... | 114 |
| Techniques for Analyzing Dumps Using IPCS REXX EXECs..... | 115 |
| Adding Hexadecimal Dump Addresses..... | 115 |
| Following a Pointer Chain..... | 115 |
| Formatting a Control Block..... | 116 |
| Customizing Control Block Analysis..... | 117 |
| Translating Dump Characters..... | 118 |
| Writing IPCS CLISTs..... | 119 |
| Controlling Output from REXX EXECs and CLISTs..... | 120 |
| FLAG Parameter..... | 120 |
| NOSUMMARY Parameter..... | 120 |
| Controlling Messages..... | 120 |
| Chapter 7. Using IPCS in Line Mode..... | 121 |
| Specifying Parameters on the IPCS Command..... | 121 |
| Setting Defaults..... | 122 |
| Invoking ISPF under IPCS..... | 122 |
| Starting the IPCS Dialog from IPCS Line Mode..... | 122 |
| Chapter 8. Using IPCS in Batch Mode..... | 125 |
| Running CLISTs and REXX Execs in Batch Mode..... | 125 |
| Running IPCS Subcommands in Batch Mode..... | 126 |
| Copying and Clearing Dumps in Batch Mode..... | 127 |
| Combining and Processing Traces in Batch Mode..... | 128 |
| Chapter 9. Printing IPCS Reports..... | 129 |
| Print and table of contents data sets..... | 129 |
| Allocating the print and table of contents data sets..... | 129 |
| Opening and closing the print and table of contents data sets..... | 130 |
| Controlling the print font..... | 131 |
| Printing into partitioned data sets | 132 |
| Allocating the partitioned data set for routing messages..... | 132 |
| Using allocation attributes for print partitioned data sets..... | 132 |
| Setting print data set report defaults..... | 133 |
| Parameters on the PROFILE subcommand..... | 133 |
| Chapter 10. Scenario to Obtain Spin Loop Diagnostic Information..... | 135 |
| Chapter 11. Examples of JCL to Print, Copy, and Clear a Dump Data Set..... | 137 |
| Sample of JCL for Printing Dumps Using IPCS..... | 137 |
| Sample of JCL for Printing Dumps into Print and Partitioned Data Sets..... | 137 |
| Example of JCL to Print, Copy, and Clear a Dump Data Set..... | 137 |
| Example of a Procedure to Print, Copy, and Clear a Dump Data Set..... | 138 |

| | |
|---|------------|
| Chapter 12. Managing Problems and Data Sets..... | 143 |
| Managing Problems..... | 143 |
| Allocating a Problem Directory..... | 143 |
| Problem Attributes..... | 145 |
| Managing Data Sets Associated with Problems..... | 146 |
| Allocating a Data Set Directory..... | 146 |
| Attributes of Data Sets..... | 147 |
| Chapter 13. Managing Dump and Trace Data Sets..... | 149 |
| Considerations for Managing the Data Sets..... | 149 |
| Data Sets with Sequential or Direct Organization..... | 149 |
| Data Sets with Partitioned Organization..... | 149 |
| Dump Data Set Initialization..... | 150 |
| Reblocking Dump Data Sets..... | 150 |
| Appendix A. Accessibility..... | 151 |
| Notices..... | 153 |
| Terms and conditions for product documentation..... | 154 |
| IBM Online Privacy Statement..... | 155 |
| Policy for unsupported hardware..... | 155 |
| Minimum supported hardware..... | 155 |
| Trademarks..... | 156 |
| Index..... | 157 |

Figures

| | |
|--|----|
| 1. Overview of IPCS..... | 4 |
| 2. IPCS as a TSO/E Application..... | 9 |
| 3. IPCS Primary Option Menu..... | 14 |
| 4. Specifying the Dump Data Set Using Option 0 (DEFAULTS)..... | 17 |
| 5. IPCS Default Values Panel Showing Update Indication..... | 17 |
| 6. IPCS Primary Option Menu Panel..... | 23 |
| 7. IPCS Default Values Panel..... | 25 |
| 8. IPCS Browse Option Entry Panel..... | 26 |
| 9. IPCS MVS Analysis of Dump Contents Panel..... | 26 |
| 10. IPCS MVS Dump Component Data Analysis Panel..... | 28 |
| 11. IPCS Trace Processing Panel..... | 30 |
| 12. IPCS Utility Menu Panel..... | 31 |
| 13. IPCS Data Set List Utility Panel..... | 32 |
| 14. First window before the DAE Display panel..... | 33 |
| 15. DAE Display Panel..... | 33 |
| 16. DAE Display Panel..... | 35 |
| 17. SADMP DASD dump data set panel..... | 38 |
| 18. IPCS Inventory Panel..... | 38 |
| 19. IPCS MVS Dump Batch Job Option Menu Panel..... | 40 |
| 20. IPCS Subcommand Entry Panel..... | 41 |
| 21. IPCS Dialog Tutorial Panel..... | 42 |
| 22. Dump Display Reporter Panel — Result of Processing a Subcommand..... | 45 |
| 23. Overriding Defaults on the Browse Option Entry Panel..... | 46 |

| | |
|--|-----|
| 24. IPCS Dialog Pointer Panel..... | 47 |
| 25. Entering Line Commands in the Pointer Stack of the Browse Option Pointer Panel..... | 48 |
| 26. Storage Panel Displayed after Entering Line Commands in the Pointer Stack of the Browse Option Pointer Panel..... | 49 |
| 27. Selecting a Pointer on the Browse Option Pointer Panel..... | 49 |
| 28. Result of Selecting a Pointer on the Browse Option Pointer Panel..... | 50 |
| 29. IPCS Storage Panel Browse Option Pointer Panel..... | 50 |
| 30. Entering the %, H, and L Selection Codes on the Browse Option Storage Panel..... | 52 |
| 31. Result of Processing the % Selection Code on the Browse Option Storage Panel..... | 52 |
| 32. Pointer Panel Result of Processing the %, H, and L Selection Codes on the Browse Option Pointer Panel..... | 53 |
| 33. Storage Summary Comments on the Browse Option Storage Panel..... | 54 |
| 34. Using Primary Commands on the Browse Option Storage Panel..... | 55 |
| 35. Result of Processing the LOCATE Primary Command on the Browse Option Storage Panel..... | 55 |
| 36. Result of Processing the STACK Primary Command on the Browse Option Pointer Panel..... | 56 |
| 37. LISTDUMP Output in 64-bit Dump..... | 56 |
| 38. Pointer Panel for the Browse Option..... | 56 |
| 39. Displaying a 64-bit Location..... | 57 |
| 40. Using the LIST Subcommand for a 64-bit Dump..... | 57 |
| 41. Output from the LIST Subcommand - 64-bit Dump - Example 1..... | 58 |
| 42. Output from the LIST Subcommand - 64-bit Dump - Example 2..... | 58 |
| 43. Using the OPCODE Subcommand..... | 59 |
| 44. Output from the OPCODE Subcommand..... | 59 |
| 45. Data Privacy for Diagnostics Usage Cycle..... | 64 |
| 46. analysis_config.json example..... | 71 |
| 47. Sample CLIST to Refresh a Dump Directory..... | 89 |
| 48. Example of Starting an IPCS Session from a REXX EXEC..... | 105 |

| | |
|---|-----|
| 49. Example of Starting an IPCS Session from a CLIST..... | 105 |
| 50. Changing the Host Command Environment to IPCS for All Commands..... | 109 |
| 51. Changing the Host Command Environment to IPCS for a Single Command..... | 110 |
| 52. Using SIGNAL ON HALT and TRACE OFF and Checking Return Code..... | 110 |
| 53. Example of a REXX EXEC Function to Retrieve Dump Data..... | 112 |
| 54. Example of a REXX Function to Display Output..... | 113 |
| 55. Sample IPCS REXX EXEC to Copy Dump Directory Information..... | 114 |
| 56. A Function to Add Hexadecimal Dump Addresses..... | 115 |
| 57. Example of a REXX EXEC Function to Follow a TCB Structure..... | 116 |
| 58. Example of a REXX EXEC Function to Format the RTM2WA..... | 117 |
| 59. Example of REXX Code to Analyze the Contents of a TCB..... | 118 |
| 60. A REXX EXEC Function for Translating Unprintable Dump Characters..... | 118 |
| 61. Running the BLSCSCAN CLIST in Batch Mode..... | 126 |
| 62. Running a Subcommand in Batch Mode..... | 127 |
| 63. Running the COPYDUMP Subcommand in Batch Mode..... | 127 |
| 64. Copying and Clearing a SYS1.DUMPnn Data Set in Batch Mode..... | 127 |
| 65. Example JCL to Print a Dump with IPCS..... | 137 |
| 66. Example JCL to Print a Dump with IPCS into Print and Partitioned Data Sets..... | 137 |
| 67. Example JCL to Run IPCS for Printing, Copying and Clearing a Dump Data Set..... | 138 |
| 68. IPCS Procedure to Print, Copy and Clear a Dump Data Set..... | 139 |
| 69. JCL to Invoke an IPCS Procedure to Print, Copy and Clear a Dump Data Set..... | 139 |
| 70. START Command Example to Print, Copy and Clear a Dump..... | 139 |
| 71. IPCS CLIST Used to Print, Copy and Clear a Dump Data Set..... | 140 |

Tables

- 1. Libraries that contain IPCS code..... 9
- 2. Options are shipped with IPCS and corresponding subcommands..... 28
- 3. Line commands for data set list..... 32
- 4. Types of symptom strings 39
- 5. Considerations for designing and invoking IPCS CLISTs..... 119
- 6. Allocation attributes for print and TOC data sets..... 129
- 7. Allocation attributes for print partitioned data sets..... 132

About this information

This information describes how to use the interactive problem control system (IPCS). The information explains how to start and use IPCS to analyze dumps and traces, and describes various functions and facilities that are available through IPCS.

This information does not try to explain MVS system concepts, such as error recovery or dumping and tracing services. Instead see the following information for that material:

- [z/OS MVS Diagnosis: Tools and Service Aids](#) describes how to obtain dumps and traces and how to analyze them.
- [z/OS Problem Management](#) gives procedures for diagnosing problems on MVS systems.

Who should use this information

This information is for anyone who performs diagnosis of software on an MVS system. Usually, this person is a systems programmer.

This information stresses the use of IPCS as an aid in dump and trace analysis, and assumes that the reader:

- Understands basic MVS system concepts.
- Can code JCL statements to run programs or cataloged procedures.
- Can code in assembler language and read assembler, loader, and linkage editor output.
- Understands commonly-used diagnostic tasks.

The audience for this information ranges from users who have no previous experience with IPCS to those who have extensive IPCS experience. Because of this wide range in experience level, this information is written with two groups of users in mind:

- Users Who Have Little or No IPCS Experience

Users in this group need to understand what IPCS does and learn how to begin using IPCS to diagnose problems. If you are new, or nearly new, to IPCS, you might want to concentrate on [Part 1, “Getting started with IPCS,”](#) on page 1.

- Users Who Are Experienced with IPCS

Experienced users already know the basics of starting and using IPCS. Users in this group need information on some of the more sophisticated aspects of IPCS, so that they can obtain maximum benefit from its use. If you are an experienced user, you might want to browse [Part 1, “Getting started with IPCS,”](#) on page 1 and concentrate on [Part 2, “Using IPCS functions,”](#) on page 21.

z/OS information

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

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

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

- None.

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.

March 2025 refresh

- “Option 3.5 — DAE” on page 33 is updated with changed panels and a new panel.
- In support of OA67290: NEW FUNCTION: Data Privacy for Diagnostics Semeru 21 Support (www.ibm.com/support/pages/apar/OA67290), a note is added to the JAVA OPTIONS function in “Using Data Privacy for Diagnostics” on page 62. (APAR OA67290, which applies to z/OS 3.1 only)

November 2023 refresh

- In support of OA64204: NEW FUNCTION for Data Privacy for Diagnostics INGEST (www.ibm.com/support/pages/apar/OA64204), Db2® SQL query and CSV support are added for INGEST. The following topics are updated:
 - “Using Data Privacy for Diagnostics” on page 62.

- [“Requesting an ANALYZE run” on page 67.](#)
 - [“Requesting a REPORT run” on page 74.](#)
 - [“Requesting an INGEST run” on page 75.](#)
 - [“Requesting an EXTRACT run” on page 80.](#)
- (APAR OA64204, which applies to z/OS V2R4 and later)

Part 1. Getting started with IPCS

The information in this part is intended for new or relatively inexperienced IPCS users. The information in this part also helps you understand what IPCS does and how to start it, so you can use it to diagnose software problems.

Chapter 1. Introduction to IPCS

The interactive problem control system (IPCS) is a tool provided in the MVS system to aid in diagnosing software failures. IPCS provides formatting and analysis support for dumps and traces produced by MVS, other program products, and applications that run on MVS.

This section contains:

- [“What You Can Do with IPCS” on page 3](#)
- [“IPCS Concepts” on page 4](#)
- [“How IPCS Works as a TSO/E Application” on page 9](#)
- [“Creating IPCS Analysis and Formatting Routines” on page 11](#)

What You Can Do with IPCS

Analyzing Dumps

Dumps produced by an MVS system fall into two categories:

- Formatted dumps: SYSABEND and SYSUDUMP ABEND dumps and SNAP dumps. IPCS cannot be used with formatted dumps.
- Unformatted dumps: SVC dumps, SYSMDUMP ABEND dumps, and stand-alone dumps. IPCS formats and analyzes unformatted dumps.

When you submit unformatted dump data sets to IPCS, it simulates dynamic address translation (DAT) and other storage management functions to recreate the system environment at the time of the dump. IPCS reads the unformatted dump data and translates it into words. For example, IPCS can identify the following:

- Jobs with error return codes
- Resource contention in the system
- Control block overlays.

IPCS also helps your own dump analysis. For example, you can:

- Format control blocks. IPCS inserts field names into the output and displays the data in columns by field.
- Browse unformatted dump storage. IPCS allows you to easily follow pointers to other locations in the dump. It also retains addresses of certain locations in the dump.
- Reduce the size of a stand-alone dump. You can reduce the size of a stand-alone dump as you transfer it from tape to a direct access storage device (DASD) for IPCS processing.

Analyzing Traces

MVS has a number of tracing facilities. The system saves these traces in storage buffers and can, for traces from the generalized tracing facility (GTF) or component trace, write the trace records to data sets.

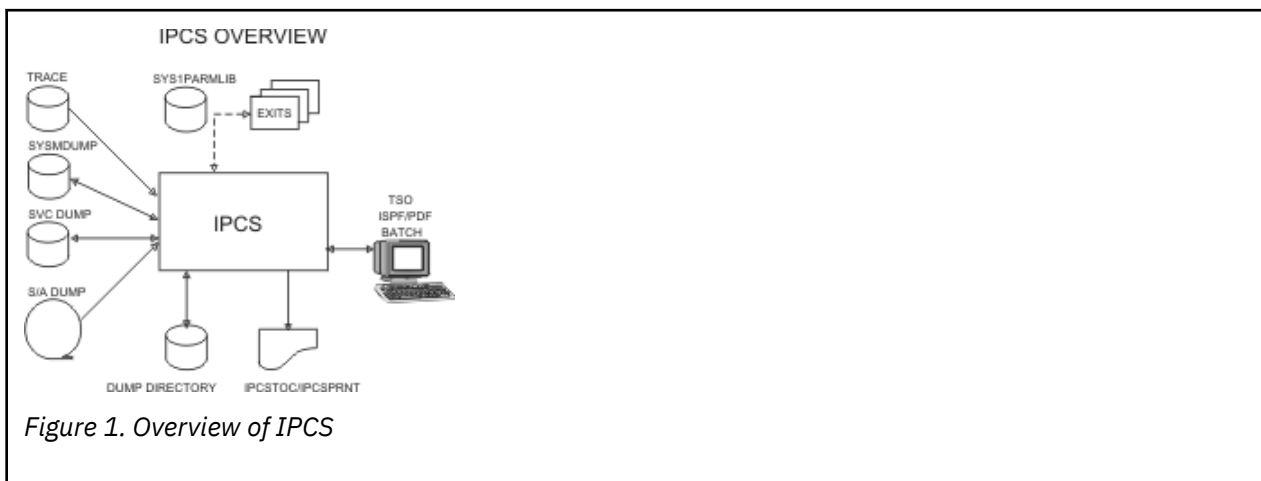
Using IPCS you can format the entries of any trace in a dump or trace data set. You can also do the following with GTF and component trace records:

- Selectively format records without deleting the unformatted data from the buffer or dump
- Find the system and time stamp for each record.
- Mix formatted GTF and component trace records without combining the unformatted data.
- Reduce the number of records in a trace data set
- Extract trace buffers from dumps

- Combine a GTF or component trace records into a single data set from multiple trace data sets.

IPCS Concepts

Figure 1 on page 4 gives an overview of how IPCS works.



IPCS accepts unformatted dumps and traces as input. It uses members of parmlib to locate control block models and exit routines. The models and routines allow IPCS to format and analyze the contents of the dump or trace data set. IPCS produces three types of output: symptom strings placed in the headers of dump data sets, control block symbols and mappings of storage saved in a dump directory, and formatted reports that the user can view at a terminal or print.

The data sets that can be directly accessed by IPCS are shown to the left and below the IPCS box in Figure 1 on page 4. As of z/OS V1R11, all these data sets are supported in cylinder-managed space. The following table gives more details.

| Data sets that can be accessed by IPCS directly | Comments, regarding placement in cylinder-managed space |
|---|---|
| TRACE data set | TRACE data sets for wrap-type traces stored in non-VSAM data sets must be initially recorded in track-managed space. IPCS, however, supports the use of COPYTRC to transcribe completed traces of this type into data sets in cylinder-managed space. |
| SVC dump data set | |
| Abend dump data set | |
| Stand-alone dump data set | DASD data sets directly used by SADMP must initially reside in track-managed space. IPCS, however, supports the use of COPYDUMP to transcribe stand-alone dumps. |
| Dump directory (VSAM data set) | |
| IPCSTOC and IPCSPRNT | In addition to DASD and tape storage devices, IPCSTOC and IPCSPRNT files can be directed to hardcopy printed destinations. |

Processing Dumps, Traces, and Other Sources

IPCS processes information from dumps or traces stored in data sets on tape or direct access storage devices (DASD) or from active system storage. The data sets and active storage are sources for IPCS processing. The type of processing IPCS performs depends on the source. IPCS processes the following sources:

- Data sets for dumps:

- SVC dumps written to automatically allocated dump data sets or to SYS1.DUMPnn data sets on DASD or tapes
- ABEND dumps written to data sets defined by SYSMDUMP DD statements
- Stand-alone dumps produced by the AMDSADMP service aid
- Component traces or generalized trace facility (GTF) traces in dumps or in trace data sets
- Unformatted information in virtual storage access method (VSAM) data sets or other types of data sets
- Active system storage for the IPCS user's address space

See [z/OS MVS Diagnosis: Tools and Service Aids](#) for more information about dumps and traces.

When you first access a source, IPCS recognizes the type of data in the source and begins the appropriate type of processing.

Dump Data Set Processing

IPCS decides if the source data set should be treated as a system dump by comparing the data set to the following criteria:

- The dump data must be stored on a data set with sequential (PS), direct (DA), or unidentified (*) organization. With z/OS R2 and later, IPCS also allows data stored on z/OS UNIX file systems to be accessed.
- The logical record length (LRECL) must be 4160 bytes.
- The data set must have one of the following combinations of record format (RECFM) and block size (BLKSIZE):
 - RECFM=F, BLKSIZE=4160
 - RECFM=FB, BLKSIZE=4160
 - RECFM=FBS, BLKSIZE=n*4160 where n=1,2,...

If the data set meets these criteria, IPCS provides dump processing, meaning that IPCS simulates system services, such as dynamic address translation and control block formatting, when processing the source.

Files stored the z/OS UNIX file system that may contain dumps for access by IPCS users, can be accessed by their path names. The size of the z/OS UNIX file must be a multiple of 4160 bytes. There are two methods by which IPCS can access these files — the FILE keyword and the PATH keyword.

- FILE support is extended in z/OS Release 2 to allow the association of a z/OS *ddname* with a full path name of from 3 to 255 characters. IPCS uses the *ddname* to access the file.

```
ALLOCATE FILE(MYDD) PATH('/u/x')
OPEN FILE(MYDD) DEFAULT
```

This support is available also for systems at OS/390® Release 10 and higher with APAR OW44412.

- PATH support is added in z/OS Release 2. IPCS accepts any valid path name up to 44 characters in length. The limit is applied after implicit qualifiers, if any, are resolved.

```
OPEN PATH('/u/x') DEFAULT
```

If partially-qualified path names are used, IPCS will determine the fully-qualified path names.

Dumps that are in extended format data sets instead of basic or large format data sets have these advantages:

- Have a greater capacity than sequential data sets.
- Support striping.
- Support compression.

Some dump data sets are quite large compared with other data sets generated by a system. The capacity of an extended format data sets is enough to hold the largest stand-alone dumps, as much as 128 gigabytes.

Striping spreads sections, or *stripes*, of a data set across multiple volumes and uses independent paths, if available, to those volumes. The multiple volumes and independent paths accelerate sequential reading and writing of the data set, reducing the time during which dump and trace I/O competes with production I/O.

In a striped data set, when the last volume receives a stripe, the next stripes are placed on the first volume, the second volume, the third, and so on to the last volume, then back to the first volume. If n volumes are used, striping allows sequential access to the data set at nearly n times the rate at which a single volume data set can be processed. The faster processing speeds up moving dump data from relatively expensive data space storage to less expensive DASD.

Compression allows dump data sets to use less DASD space. Before using compression, however, remember that compression and decompression trade off processing cycles for more efficient use of DASD. If software compression is used because hardware compression is not available, the number of processing cycles is significantly higher and noticeably impacts response time.

Trace Data Set Processing

These data sets have certain requirements, which are described in [*z/OS MVS Diagnosis: Tools and Service Aids*](#). You can use the IPCS subcommands that apply to component traces or GTF traces when formatting the trace records.

VSAM and Other Data Set Processing

When IPCS does not consider the source to be a system dump or trace data set, IPCS provides general purpose processing. In this processing, IPCS initializes the data set, processes records in the data set, and makes the unformatted records available for accessing and viewing. However, you cannot use IPCS subcommands for these data sets.

Active Storage Processing

IPCS can process as a dump the central storage for the address space in which IPCS is currently running, private storage, and any common storage accessible by an unauthorized problem state program. Users running z/OS R2 IPCS on a z/OS R2 system who have been authorized READ access to facility class resource BLSACTV.ADDRSPAC may view storage that is fetch-protected from application code.

IPCS does not create a storage map when processing active storage, but it does maintain a limited symbol table. Limitations on IPCS subcommands exist when processing active storage.

Using IPCS at the Terminal

IPCS has three modes of operation:

- Line mode on a terminal
- Full-screen mode on a terminal, known as the *IPCS dialog*
- Batch mode using the terminal monitor program (TMP)

The following sections summarize these modes.

IPCS Line Mode

IPCS is a TSO/E command processor. When you enter the IPCS command at the TSO/E READY prompt, you begin an IPCS line mode session. In this mode you can enter IPCS subcommands and invoke REXX execs and CLISTs. IPCS provides online explanations of the function, syntax, and parameters of each IPCS subcommand.

IPCS Dialog

IPCS provides a full-screen dialog through the Interactive System Productivity Facility (ISPF). This mode, known as the IPCS dialog, runs on top of ISPF and the IPCS command processor. The IPCS dialog provides the following advantages over line mode:

- Organizes the approach to diagnosis
- Provides easy-to-use panels for selecting dump analysis, formatting, and management procedures
- Enhances the functions available when browsing unformatted storage records
- Captures analysis reports for easier viewing
- Offers online help for reading analysis reports and a tutorial on using the IPCS dialog.

To process a source in the IPCS dialog, specify the data set as the source on the IPCS default panel, in a SETDEF subcommand, or in a parameter on a subcommand. If the source is active storage, specify the MAIN parameter.

Each IPCS panel has its own local defaults, including the source. The dialog facilities, subcommands, and command procedures used on the panel use the same source as the panel. Another panel, with its subcommands and command procedures, can use a different set of local defaults, including a different source. This arrangement allows you to view several, related dumps in parallel, which is helpful in diagnosing problems on different systems in a sysplex. Commands entered from a window run on the dump being viewed in the window.

IPCS Batch Mode

You can submit batch jobs that access IPCS to perform dump analysis against a specific dump.

Saving Dump Analysis Results

As you examine a dump of a software failure, IPCS accumulates information about the dump. IPCS uses the information it gathers each time you process the dump. IPCS saves the results of dump analysis.

User Dump Directory

The majority of the information generated by dump analysis is stored in the user dump directory, which is a data set allocated to the ddname IPCSDDIR. The user dump directory contains information, called a *source description*, used by IPCS to process dumps, traces, and other sources efficiently.

The source descriptions resemble a notebook that IPCS uses as it processes a source; IPCS writes notes about the dump, such as its symbols, the location of control blocks, and session defaults. Later, when IPCS needs some information from its notes to format a dump, IPCS obtains it from the source description for the dump.

Sysplex Dump Directory

In a sysplex, the systems can share a sysplex dump directory. This directory contains source descriptions like the ones in a user dump directory, but the source descriptions are for sources from all the systems that share the sysplex dump directory.

When any system that shares the sysplex dump directory produces an SVC dump, the system automatically writes a source description for the dump in the sysplex dump directory. If you want to analyze the dump, you can copy the source description from the sysplex dump directory to your user dump directory.

Dump Header Record

When IPCS identifies a specific problem in the dump, IPCS might add a symptom to the dump header record. These symptoms are added to the dump to make them available to the IBM Support Center in case the center receives the dump for an authorized problem analysis report (APAR). For complete information on symptoms, see [z/OS Problem Management](#).

Print Data Set

You have the option of using IPCS interactively or printing reports. You override a default parameter to tell IPCS to send its output to the IPCSPRNT data set.

Providing Format and Analysis Support

IPCS uses members of SYS1.PARMLIB or parmlib members that make up a logical parmlib concatenation to control its processing:

- IPCSPRxx
- BLSCECT
- BLSCUSER, which is embedded in the default BLSCECT member
- Other members embedded in BLSCECT

Prior to the concatenated parmlib support available with OS/390 Release 2, IPCS used an IPCSPARM data set to identify the members required for its processing. If an IPCSPARM data set was not allocated at the time IPCS was invoked, IPCS dynamically allocated SYS1.PARMLIB for the duration of the IPCS session and unallocated it when complete.

If IPCS is running on a level of z/OS that supports concatenated parmlib and if an IPCSPARM data set is not allocated, IPCS allocates the current concatenation available and frees it when complete.

IPCSPRxx Parmlib Member

IPCS has a set of parameters that control the problem management and data set management functions and the default sizes of the print data set to be used during the IPCS session. The IPCSPRxx parmlib members contain defaults for these parameters.

IPCSPR00, the IBM-supplied member, suppresses the use of problem and data set management through the NODSD and NOPDR parameters. To use these facilities, create new IPCSPRxx members and specify the DSD and PDR parameters.

BLSCECT Parmlib Member

BLSCECT, the exit control member, defines the dump analysis and formatting models and verb exit routines that are available to IPCS. See [z/OS MVS Initialization and Tuning Reference](#) for the default contents of BLSCECT.

BLSCECT also embeds parmlib members that provide additional formatting and analysis support to IPCS.

BLSCUSER Parmlib Member

IBM recommends that you create a BLSCUSER member to define installation-supplied exits. By default, BLSCECT embeds BLSCUSER.

IBM recommends that you do not modify BLSCECT. If you do modify BLSCECT, you will need to allocate the modified BLSCECT to the IPCSPARM data set when you start an IPCS session. Also, you must carry the installation modifications to the new BLSCECT for the next release.

For example, if you have installation-defined dump exits that were previously in MVS/XA SP load module AMDRECT, you can add them to BLSCUSER.

Other Pieces of IPCS

PRDMP

Print dump (PRDMP) is a service aid that provides functions available through IPCS subcommands and exit routines. An appendix in [z/OS MVS IPCS Commands](#) gives the equivalent PRDMP functions in IPCS.

Problem and Data Set Management

IPCS has facilities to keep track of problems with your system and dump data sets stored on the system. However, you usually do not need to use these facilities. The Information/System family of products provides the same function.

How IPCS Works as a TSO/E Application

IPCS is a problem-state, key 8 program that runs in a TSO/E user's address space. IPCS operates in interactive and batch environments supported by TSO/E.

The base of IPCS is a TSO/E command processor. The TSO/E command “IPCS” activates the IPCS command processor. All commands used to perform IPCS functions are “subcommands” of the IPCS command. You can use IPCS functions from any TSO/E line mode session.

For interactive use, the IPCS dialog uses ISPF dialog support to run as an interactive, full-screen application. This application uses the IPCS command processor. z/OS R2 IPCS exploits data spaces, if permitted, to free virtual storage to allow large, complex analysis routines to function.

Figure 2 on page 9 shows where the IPCS command processor and IPCS dialog fit in the system. The IPCS command processor environment is not available from ISPF; you cannot invoke IPCS subcommands directly from the ISPF dialog.

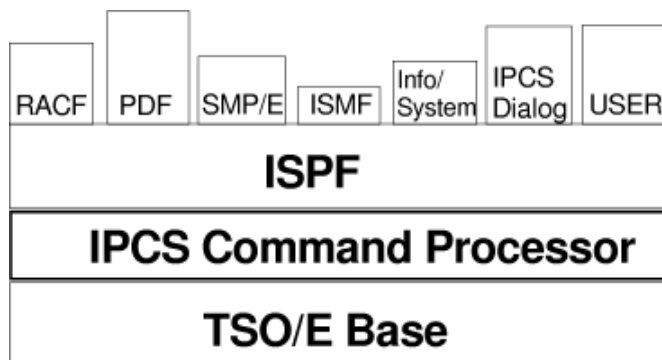


Figure 2. IPCS as a TSO/E Application

A set of data set libraries contain the code to run IPCS. [Table 1 on page 9](#) lists these data sets.

| Table 1. Libraries that contain IPCS code | |
|---|---|
| Data Set | Contents |
| SYS1.HELP | TSO/E HELP data for IPCS commands and subcommands |
| SYS1.MIGLIB | IPCS code |
| SYS1.PARMLIB | IPCS parmlib members |
| SYS1.SBLSCLI0 | IPCS REXX execs and CLISTs |
| SYS1.SBLSKELO | ISPF skeletons for the IPCS dialog |
| SYS1.SBLSMSG0 | ISPF messages for the IPCS dialog |
| SYS1.SBLSPNLO | ISPF panels for the IPCS dialog |
| SYS1.SBLSTBLO | ISPF tables for the IPCS dialog |

IPCS Command Processing

IPCS processes commands, subcommands, CLISTs, and REXX execs.

TSO/E Commands for IPCS

IPCS provides three commands to be invoked from the TSO/E READY prompt. They are:

- IPCS
- IPCSDDIR
- SYSDSCAN.

Other TSO/E commands may have unique processing features when issued from an IPCS dialog session. See [z/OS MVS IPCS Commands](#) for information about these commands.

IPCS Subcommands

Once you enter the IPCS command to begin an IPCS session, the IPCS subcommands are your main tools for performing dump and trace analysis. These commands allow you to analyze, format, view, retrieve, and copy dump and trace data, and to maintain an IPCS session. You may use subcommands in any mode.

Subcommands are an extension to the TSO command language and can be as long as 32000 characters in length. Common TSO language verbs like LIST (alias L) are used to elicit information displays.

IPCS Primary and Line Commands

Another set of IPCS commands are available for use in the full-screen dialog. These commands control various panel functions. The primary commands are entered on the COMMAND or OPTION line of the IPCS dialog. The line commands are used in the prefix area of an IPCS dialog.

Primary commands are an extension of the ISPF dialog language and are generally limited in length to a few less characters than your screen width. Common ISPF verbs like LOCATE (alias LOC, LIST or L) are used to scroll within data based on key information shown on panels that support scrolling. To access IPCS subcommands from any panel of the IPCS dialog, use the IPCS primary command as a prefix.

REXX Execs and CLISTs

You can invoke REXX execs and CLISTs from an IPCS session. These procedures can enter subcommands or use other REXX and CLIST functions to analyze dumps and traces. IPCS provides functions to store data in REXX or CLIST variables and to print data to the IPCS dialog or print data set.

Running Different Levels of IPCS

The version and release level of IPCS must match the level of the system that produced the dump. You must use the z/OS MVS libraries of IPCS code, for example, to analyze a dump or trace produced by an z/OS MVS system.

With z/OS Release 2, enhancements are added that make it more important that the user confirm that the level of IPCS being used matches the level of the system that generated dumps and traces. If there is a mismatch between the IPCS level and the level of the dumping program, IPCS will issue an appropriate message when IPCS dump initialization is performed.

- If the levels match, message BLS18223I is issued; *source* is one of the following: SADUMP, SVC, SYSDUMP, SLIP, or unknown agent.

```
BLS18224I Dump of z/OS 01.13.00-0 - level differs from
          IPCS level
BLS21001I IPCS for z/OS 01.13.00-1
```

- If there is a mismatch between the level of the dumped system and the IPCS level, message BLS21001I follows message BLS18224I.

In all cases, IPCS will attempt to process the dump despite the lack of information about the level of system being dumped.

Also with z/OS Release 2, IPCS issues messages to indicate whether the level of IPCS matches the level of the dumping program and the level of the system being dumped. If all are at the same level, one of the following messages will be issued:

```
BLS18224I Dump of z/OS 01.02.00 -  
          level same as IPCS level  
BLS18224I Dump of z/OS 01.02.00 -  
          level same as both SADUMP  
          and IPCS levels
```

With z/OS with FMID JBB778H, an additional indicator is added to identify an MVS release. IPCS has been updated to display this indicator as a part of the product name as shown below:

```
BLS18224I Dump of z/OS 01.13.00-0 - level differs from  
          IPCS level  
BLS21001I IPCS for z/OS 01.13.00-1
```

With the exception of SADUMP, z/OS dumping programs are part of the system itself and cannot be matched with the wrong system level. SADUMP is intended to be used to process a system of a matching level. However, if there is a mismatch between the level of the dumped system and the SADUMP level, message BLS18224I identifies the nature of the mismatch.

- If there is a mismatch between the level of the dumped system and the SADUMP level, message BLS18224I identifies the level of SADUMP used to generate the dump.
- If there is a mismatch between the level of the dumped system and the IPCS level, message BLS2100I precedes message BLS18224I.

In all cases, IPCS will attempt to process the dump.

Although IPCS and the dump or trace must have matching levels, you can run IPCS on different levels of the system. See [z/OS MVS IPCS Customization](#) for requirements for running IPCS on different levels of z/OS.

IPCS Messages and User Completion Codes

IPCS issues its own set of TSO/E messages. These messages identify problems with the IPCS session or problems with the data in a dump or trace data set. See [z/OS MVS Dump Output Messages](#) for explanations.

If an abend occurs during IPCS processing, IPCS ends with a completion code. You can find these codes and the procedure for diagnosing the problems in [z/OS MVS Diagnosis: Reference](#).

Creating IPCS Analysis and Formatting Routines

You can create your own formatting support using REXX execs, CLISTs, or assembler language exit routines. This book explains how to write REXX execs and CLISTs for IPCS. See [z/OS MVS IPCS Customization](#) for information about creating exit routines.

Chapter 2. Starting IPCS

Before you can use IPCS to analyze dumps or traces, you must start IPCS and specify the source of the information you want to analyze. You might also need to respond to messages that the system issues during dump or trace data set initialization. Then you need to understand how to request IPCS functions.

Starting IPCS

IBM recommends that installations provide customized access to IPCS. Customized access allows you, as a user of IPCS, to quickly and easily start IPCS in a way that is consistent with the installation's standards.

Starting IPCS without customized access can be a complex task. This process can consist of many steps, such as:

- Concatenating the IPCS libraries
- Allocating an alternate parmlib data set
- Creating, initializing, and allocating the dump directory data set
- Starting IPCS line mode
- Starting the ISPF dialog
- Starting the IPCS dialog

It is possible for you to do these things yourself each time you use IPCS, but starting IPCS is much simpler when customized access is available.

Note: This section is primarily concerned with using IPCS in the full-screen mode known as the IPCS dialog. For information on using IPCS in line mode or batch mode see [Chapter 7, “Using IPCS in Line Mode,”](#) on page 121 and [Chapter 8, “Using IPCS in Batch Mode,”](#) on page 125.

Customized Access

Much of the process of starting IPCS can be automated through customized access. The customization recommended by IBM provides an option for starting the IPCS dialog from an ISPF panel; the user simply selects the appropriate option. See [“Starting IPCS with Customized Access”](#) on page 13 for a description of this method.

If you do not know which ISPF panel provides an option for IPCS, or you are not sure what customization your installation has done, contact the system programmer responsible for customizing IPCS for users at the installation. The system programmer should be able to describe any customization that has been done and explain any installation-specific procedures.

At some installations, there might not be a system programmer responsible for customizing IPCS. In this case, you have two choices:

- See *z/OS MVS IPCS Customization* for information about customizing access to IPCS, and follow the steps described there. This approach is preferred. After you have customized access, you can take advantage of this customization every time you use IPCS.
- Follow the procedure described in [“Starting IPCS without Customized Access”](#) on page 14. This method is the simplest way to start IPCS without customized access. However, until customized access is available, you would have to use this procedure every time you start IPCS.

Starting IPCS with Customized Access

As part of customizing access to IPCS, IBM recommends that you or your installation provide an option for starting the IPCS dialog from an ISPF selection panel, usually the ISPF Primary Option Menu. To start the IPCS dialog from such an ISPF panel, select the option for IPCS.

After you select the IPCS option and press ENTER, the system displays the IPCS Primary Option Menu, as shown in [Figure 3 on page 14](#).

```
----- IPCS PRIMARY OPTION MENU -----
OPTION  ==>

0  DEFAULTS  - Specify default dump and options
1  BROWSE    - Browse dump data set
2  ANALYSIS  - Analyze dump contents
3  UTILITY   - Perform utility functions
4  INVENTORY - Inventory of problem data
5  SUBMIT    - Submit problem analysis job to batch
6  COMMAND   - Enter subcommand, CLIST or REXX exec
T  TUTORIAL  - Learn how to use the IPCS dialog
X  EXIT      - Terminate using log and list defaults

*****
* USERID - IPCSU1
* DATE   - 95/03/12
* JULIAN  - 95.072
* TIME    - 21:51
* PREFIX  - IPCSU1
* TERMINAL - 3278
* PF KEYS - 12
*****

Enter END command to terminate IPCS dialog
```

Figure 3. IPCS Primary Option Menu

At this point, you have started the IPCS dialog. Next, you need to specify the data set that contains the data you want to analyze. See [“Setting Session Defaults” on page 15](#) for instructions.

If the system does not display the IPCS Primary Option Menu, it is usually an indication that the IPCS libraries have not been properly concatenated. If you have customized access to IPCS yourself, make sure that you have correctly followed the procedures described in [z/OS MVS IPCS Customization](#). Otherwise, report the problem to the system programmer responsible for customizing IPCS at your installation.

Starting IPCS without Customized Access

This method is recommended only when neither you nor your installation has customized access to IPCS. This method uses the IPCS defaults for creating a dump directory and allocating IPCS data sets. This method does not allocate the IPCS print data set or activate the problem and data set management functions.

To start IPCS, do the following:

1. **Logon to TSO/E.**
2. **(Optional Step):**
 - a. **Add SYS1.SBLSCLIO or a copy of it to your SYSPROC data set concatenation** by entering the following command:

```
ALTLIB ACTIVATE APPLICATION(CLIST) DA('SYS1.SBLSCLIO')
```

- b. **Start IPCS line mode** by entering the IPCS command:

```
IPCS
```

Most users, including those who are new to IPCS, should skip this step and proceed directly to step [“3” on page 15](#). However, you should perform this step if you plan to:

- Use IPCS in line mode
- Use the problem management and data management subcommands
- Enter and exit the IPCS dialog frequently during an ISPF session.

In the first two cases, it is necessary to enter the IPCS command. In the last case, entering the IPCS command is not necessary, but results in improved response time when entering and exiting the IPCS dialog.

See [“Specifying Parameters on the IPCS Command” on page 121](#) for further information about parameters you can use with the IPCS command.

After you have entered the IPCS command, an IPCS prompt replaces the READY prompt. At this point, you can use IPCS in line mode. You do not need to proceed to the next step unless you want to start the IPCS dialog from IPCS line mode.

3. **Start the ISPF dialog** by entering the ISPF command:

```
ISPF
```

Successful processing displays the ISPF/PDF Primary Option Menu (or installation-provided menu).

4. **Choose option 6 or TSO/E commands** from the ISPF menu:

```
----- ISPF PRIMARY OPTION MENU -----  
OPTION  ==> 6
```

5. **Start the IPCS dialog** by entering the following command at the prompt:

```
EX 'SYS1.SBLSCLI0(BLSCLIBD) '
```

The BLSCLIBD CLIST starts the IPCS dialog. Next, proceed to [“Setting Session Defaults” on page 15](#) for information about establishing defaults for your IPCS session.

Setting Session Defaults

After you have started IPCS, the next step is to set the source default value to specify the source of the data you want to analyze. The source is usually the only default value you want to change, but, you can change other defaults. This section discusses the IPCS session defaults and explains how to change the source and other default values.

IPCS Default Values

IPCS is shipped with certain session default values. When you begin your first IPCS session, or whenever you start a session with a new user dump directory, IPCS establishes these defaults. See the SETDEF subcommand in [z/OS MVS IPCS Commands](#) for a list of the default values supplied with IPCS.

IPCS maintains your default values from one session to the next. Thus, the defaults in effect at the end of your IPCS session will still be in effect the next time you start IPCS, provided that you use the same user dump directory. These defaults remain in effect until you explicitly change them.

By taking the time to set the session defaults when you begin your first IPCS session, you can avoid having to specify them on each subcommand to which they apply.

Local and Global Defaults

IPCS has two types of default values:

- Local defaults. These values are currently in use for an ISPF screen in the IPCS dialog, for a batch IPCS session, or for an IPCS interactive line-mode session.
- Global defaults. These values are used to establish the local defaults when IPCS processing starts in an ISPF screen, a batch IPCS session, or an IPCS interactive line-mode session.

Your global defaults are obtained from the dump directory being used. IPCS uses as the global defaults the following, in this order:

1. The last value specified as a global default in a SETDEF subcommand or on the IPCS Default Values panel in the IPCS dialog.
2. The value in the IPCSPRxx parmlib member. For information on IPCSPRxx see [“Providing Format and Analysis Support” on page 8](#).
3. The IBM-supplied value.

In addition, a subcommand can override a current local default by specifying a SETDEF parameter. For each subcommand in [z/OS MVS IPCS Commands](#), the SETDEF-defined parameters are grouped in the syntax diagram, thereby identifying the SETDEF-defined parameters that apply specifically to the subcommand. The overriding values in these parameters apply only to the subcommand, are not saved in your user dump directory, and are not retrieved by an EVALDEF subcommand.

Defaults for IPCS Dialog Panels

Each IPCS panel and each IPCS subcommand can have its own local defaults, including the source. The local defaults for a panel are used by the dialog facilities, subcommands, and command procedures used on that panel. This arrangement allows you to view several, related dumps in parallel, which is helpful when diagnosing from multiple dumps from systems in a sysplex. The following rules apply to local defaults for panels:

- Local defaults in effect when you initiate processing are associated with that processing. The processing can consist of a dialog panel, a subcommand, or a command procedure, such as a CLIST or a REXX exec.
- When entered on a panel, commands that read the source, such as a dump, run on the source being viewed in the panel.
- When entered on a panel, commands that write to the source can change the defaults being used.

To specify local defaults for processing, use any of the following techniques:

- Split your screen and use the IPCS Default Values panel to set the local defaults for each screen.
- Start analyzing one source using IPCS dialog panels. Then, enter the IPCS Browse panel and specify a different source on it. The other IPCS dialog panels will continue showing the first source.
- Start analyzing one source using IPCS dialog panels. Then, enter a subcommand on the IPCS Commands panel and specify a different source in the subcommand. All further subcommands entered on the IPCS Commands panel will apply to the second source, but the other panels will apply to the first source.

At any time, you can enter a subcommand specifying the first source to make subcommands entered on the IPCS Commands panel apply to the first source.

PROFILE Default Values

You can specify default values for the line size and the number of lines per page for printed output through the PROFILE subcommand. These defaults are recorded in your dump directory and remain in effect until you change them with another PROFILE subcommand. See the PROFILE subcommand in [z/OS MVS IPCS Commands](#) for more information.

Address Processing Parameters

The ASID and CPU parameters are address processing parameters that are part of the data description parameter. The data description parameter is used to describe storage in a dump. The ASID and CPU parameters are null until you specify a dump data set. When you specify a dump and access it with a dump analysis subcommand, that subcommand sets the ASID and CPU parameters to describe an address space contained in that dump.

If possible, IPCS sets the ASID parameter to the address space identifier associated with the problem that caused the dump. However, it might not always be possible to do so, as in the following cases:

- When a dump is initiated through an operator command, a stand-alone dump program, or a similar external mechanism, IPCS might not be able to identify the address space associated with the failure.
- When there is truncation during recording or I/O errors during transcription, data needed to establish an appropriate default address space might not be present in a dump data set.

In such circumstances, IPCS attempts to select a usable address space from which analysis can proceed, although the address space selected might not be one in which a failure occurred.

If the dump is a stand-alone dump and if a store status operation was performed, the CPU parameter is set to the IPLed central processor number. Otherwise, the CPU parameter is set to another processor in the configuration or NOCPU is used.

SETDEF rejects any attempt to set these parameters before you specify a dump.

See [z/OS MVS IPCS Commands](#) for more information about the data description parameter and address processing parameters.

Specifying the Source

To analyze data in a source, specify the data set name as the source on the IPCS Default Values panel, in a SETDEF subcommand, or in a parameter on a subcommand. If the source is active storage, specify the MAIN parameter.

You can specify the source data set by using the IPCS Default Values panel in the IPCS dialog, as follows:

1. Select the DEFAULTS option from the IPCS Primary Option Menu, shown earlier in [Figure 3 on page 14](#). When you press ENTER, the system displays the IPCS Default Values panel.
2. Type the name of the dump or trace data set in the SOURCE field. If you have not previously defined a source data set, the value in this field will be NODSNAME when the panel first appears on your screen. NODSNAME indicates that your local defaults do not specify a source.

[Figure 4 on page 17](#) shows an example of how to specify a dump data set named D46IPCS.SVC.CSVLLA.DUMP002 as the source. You can also change any of your other local defaults at this time if you wish.

```
----- IPCS Default Values -----
COMMAND ==> _

You may change any of the defaults listed below. The defaults shown before
any changes are LOCAL. Change scope to GLOBAL to display global defaults.

Scope ==> LOCAL (LOCAL, GLOBAL, or BOTH)

If you change the Source default, IPCS will display the current default
Address Space for the new source and will ignore any data entered in
the Address Space field.

Source ==> DSNAME('D46IPCS.SVC.CSVLLA.DUMP002')
Address Space ==>
Message Routing ==> NOPRINT TERMINAL NOPDS
Message Control ==> FLAG(WARNING) CONFIRM VERIFY
Display Content ==> NOMACHINE REMARK REQUEST NOSTORAGE SYMBOL

Press ENTER to update defaults.

Use the END command to exit without an update.
```

Figure 4. Specifying the Dump Data Set Using Option 0 (DEFAULTS)

3. Press ENTER. IPCS displays UPDATED in the upper right of the panel, indicating that the local defaults have been changed.

```
----- IPCS Default Values -----UPDATED
COMMAND ==> _

You may change any of the defaults listed below. The defaults shown before
```

Figure 5. IPCS Default Values Panel Showing Update Indication

4. Press PF3 to exit the panel.

Now that you have updated the session defaults, you are ready to begin using IPCS to analyze dump or trace data.

Initializing Dumps and Traces

IPCS initializes a dump data set or trace data set when you enter the first IPCS dialog option or IPCS subcommand that performs formatting or analysis of the dump or trace.

Dump Data Set Initialization

When a dump is first accessed, IPCS:

- Initializes the dump.
- Reads the entire dump sequentially.

- Builds a description of the dump data set in the dump directory.
- Locates and records the positions of all physical blocks in the dump. From this time until a DROPDUMP subcommand removes the description from the directory, alterations to the dump data set might make the description IPCS has created obsolete.
- Creates storage map entries for the communication vector table (CVT) and certain data areas, modules, and structures pointed to from the CVT. It also creates symbol table entries for the CVT, global data area (GDA), private area, common area, and the dump title.

Summary Dump Data

If the dump contains summary dump data, IPCS issues message BLS18160D as follows:

```
BLS18160D May summary data be used by dump access? Enter Y to
use, N to bypass
```

This message asks you to indicate whether you want summary dump data processed during dump initialization. The summary dump data contains data captured closest to the time of the failure. If you reply Y to use this data, IPCS will not be able to display storage keys using the DISPLAY(MACHINE) parameter.

In some cases, the dump values generated when summary dump data is processed can be misleading. For example, if the dump is a result of the SDUMP macro and BRANCH=YES was specified, then replying:

YES

Causes the PSAAOLD field to contain the address space control block (ASCB) address of the address space that issued the SDUMP.

NO

Causes the PSAAOLD field to contain the address of either the address space that is dumped (if one ASID is being dumped) or the master scheduler's address space (if more than one ASID is being dumped).

Trace Data Set Initialization

If IPCS identifies a data set as possibly being a trace data set, it issues message BLS18099D to verify the way you want to use the data set:

```
BLS18099D Treat input only as trace data? Enter Y for yes, N for
full initialization
```

Respond **Y** if you wish to treat the input data set as a trace data set. Respond **N** if you want IPCS to begin full initialization.

For trace data sets only, IPCS bypasses normal initialization and improves the performance of trace processing by commands such as the GTFTRACE subcommand. During trace data processing, information about dumps in the dump directory is not available. Subcommands such as LISTDUMP do not work in this mode.

This mode of accessing trace data sets provides a significant performance improvement over the normal mode of access, which involves full initialization, particularly for those trace data sets residing on multi-reel tape volumes.

Note: IPCS verifies the way you want to use the data set only when the CONFIRM default value is in effect. If NOCONFIRM is in effect, IPCS automatically assumes a response of Y to message BLS18099D. CONFIRM is the default. See [“Setting Session Defaults” on page 15](#) for more information about setting defaults.

Requesting IPCS Functions

You can request IPCS functions using the IPCS dialog as follows:

- Through a selectable option in the IPCS dialog.

IPCS provides many functions directly through the IPCS dialog. You can select the option that corresponds to the function you want.

- Through an IPCS subcommand.

There are two ways to enter subcommands from the IPCS dialog:

- Choose the COMMAND option from the IPCS Primary Option Menu and enter the subcommand on the command line. For example:

```
===> STATUS WORKSHEET
```

- Use the IPCS primary command to prefix the subcommand invocation from any command or option line in the IPCS dialog. For example:

```
COMMAND ===> IPCS STATUS WORKSHEET
```

For functions available as both IPCS dialog options and IPCS subcommands, it is usually easier to select the appropriate dialog option than to issue the corresponding IPCS subcommand.

Part 2. Using IPCS functions

The information in this part helps you to take advantage of the wide range of services and facilities that IPCS offers.

Chapter 3. Using the IPCS Dialog

This section describes the IPCS full-screen dialog (referred to as the IPCS dialog throughout this publication) that is supplied with IPCS. The IPCS dialog is an interactive dialog that you use at a terminal. The IPCS dialog organizes the problem analysis process into seven options:

- Set IPCS defaults
- View formatted dump data
- Generate and edit dump analysis reports
- Submit dump analysis jobs for batch processing
- Run IPCS subcommands, CLISTs, and REXX execs
- Copy dump and trace data from one data set to another
- Manage dump and trace data set sources

The IPCS dialog also provides an extensive set of tutorials for using the dialog, specifying subcommands, and reading dump analysis reports. See [Chapter 2, “Starting IPCS,” on page 13](#) for information about starting the IPCS dialog.

IPCS Primary Option Menu

The IPCS primary option menu panel, [Figure 6 on page 23](#), first appears when you access the IPCS dialog. Note that for users running z/OS R2 IPCS and higher, the top line of the panel displays the level of IPCS being used. This allows the user to confirm that it is the same as the level of material to be analyzed.

```
----- z/OS 01.02.00 IPCS PRIMARY OPTION MENU -----
OPTION  ===>

  0  DEFAULTS      - Specify default dump and options
  1  BROWSE        - Browse dump data set
  2  ANALYSIS      - Analyze dump contents
  3  UTILITY       - Perform utility functions
  4  INVENTORY     - Inventory of problem data
  5  SUBMIT        - Submit problem analysis job to batch
  6  COMMAND       - Enter subcommand, CLIST or REXX exec
  T  TUTORIAL      - Learn how to use the IPCS dialog
  X  EXIT          - Terminate using log and list defaults

*****
* USERID - IPCSU1
* DATE   - 95/03/12
* JULIAN - 95.072
* TIME   - 21:51
* PREFIX - IPCSU1
* TERMINAL- 3278
* PF KEYS - 12
*****

Enter END command to terminate IPCS dialog
```

Figure 6. IPCS Primary Option Menu Panel

To access the IPCS dialog, see [“Starting IPCS” on page 13](#). This panel organizes all the features of the IPCS dialog into the following options:

0 DEFAULTS

Use the defaults option to review and update IPCS default values. See [“Option 0 — DEFAULTS” on page 24](#).

1 BROWSE

Use the browse option to display a requested dump in full-screen mode. See [“Option 1 — BROWSE” on page 26](#).

2 ANALYSIS

Use the analysis option to analyze the status of the MVS system and its components as they are represented in an MVS dump data set. See [“Option 2 — ANALYSIS” on page 26](#).

3 UTILITY

Use the utility option to copy dump directory information from an existing dump directory to the current dump directory, to copy dump data from one dump data set to another, to copy trace data from one data set to another, or to process a list of your source data sets. Also use the utility option

to see the symptom strings in the dump analysis and elimination (DAE) data set and to process them. See [“Option 3 — UTILITY” on page 31.](#)

4 INVENTORY

Use the inventory option to review and manage the list of dumps in your user dump directory or, if you are authorized, in the sysplex dump directory. See [“Option 4 — INVENTORY” on page 38.](#)

5 SUBMIT

Use the submit option to submit problem analysis jobs for batch processing. See [“Option 5 — SUBMIT” on page 40.](#)

6 COMMAND

Use the command option to enter subcommands, CLISTs, or REXX execs directly on the IPCS command entry panel. By using the command option, IPCS can display the output of subcommands, CLISTs, or REXX execs on a dump display reporter panel. See [“Option 6 — COMMAND” on page 41.](#)

T TUTORIAL

Use the tutorial option to see online instruction in the use of the IPCS dialog. You can view the tutorial sequentially from beginning to end, or randomly by selecting topics from its table of contents. You can also use the ISPF HELP command from any panel in the IPCS dialog. The HELP command is normally associated with program function (PF) keys 1 or 13. See [“Option T — TUTORIAL” on page 42.](#)

X EXIT

Use the exit option to leave the IPCS dialog. If you entered the IPCS dialog directly when you activated ISPF, ISPF also ends. Note that ISPF uses its defaults regarding the disposition of LIST and LOG data sets without prompting you.

The **END** primary command also ends the IPCS dialog. If you entered the IPCS dialog directly when you activated ISPF, ISPF also ends (as it does with the exit option), but ISPF prompts you to decide how to dispose of LIST and LOG data sets used during the ISPF session.

Selecting an Option

Selecting an option on a panel in the IPCS dialog is very much the same as selecting an option on an ISPF panel. Just type the number or letter of the option in the OPTION field and press Enter:

```
OPTION ==> 2_
```

Options 2, 3, or 5 on the IPCS primary option menu panel each use menu panels to provide another set of options. To bypass the second menu panel, type the number of the selection from each panel, separating them with a decimal point, on the IPCS primary option menu panel. For example, entering 2.6 on the IPCS primary option menu panel displays the Dump Component Data Analysis panel, bypassing the Analysis of Dump Contents Menu panel.

To jump from one option to another without returning to the IPCS primary option menu panel, use the jump function. Suppose you are viewing storage in the IPCS browse option and you want to generate a contention report. Enter the following in the COMMAND field of the IPCS Browse option panel:

```
COMMAND ==> =2.5_
```

This ends the browse option and displays a contention report on the dump display reporter panel.

For information on ISPF operations available in the IPCS Dialog, see [“ISPF Operations on the IPCS Dialog” on page 60.](#)

Option 0 — DEFAULTS

The IPCS Default Values panel, [Figure 7 on page 25](#), establishes the SETDEF-defined IPCS defaults to be used during an IPCS session. Invoke this panel by selecting option 0 (DEFAULTS) from the IPCS Primary Option Menu panel.

```

----- IPCS Default Values -----
COMMAND ==> _

You may change any of the defaults listed below. The defaults shown before
any changes are LOCAL. Change scope to GLOBAL to display global defaults.

Scope ==> LOCAL (LOCAL, GLOBAL, or BOTH)

If you change the Source default, IPCS will display the current default
Address Space for the new source and will ignore any data entered in
the Address Space field.

Source ==> NODSNAME
Address Space ==>
Message Routing ==> NOPRINT TERMINAL NOPDS
Message Control ==> FLAG(WARNING) CONFIRM VERIFY
Display Content ==> NOMACHINE REMARK REQUEST NOSTORAGE SYMBOL

Press ENTER to update defaults.

Use the END command to exit without an update.

```

Figure 7. IPCS Default Values Panel

This panel displays the current SETDEF-defined IPCS defaults that are stored in your user dump directory. The fields in Figure 7 on page 25 show the SETDEF-defined defaults when you first access IPCS. To change the SETDEF-defined defaults, enter the requested parameter or value on the panel. For information on the fields to be specified, see [“Setting Session Defaults” on page 15](#) or invoke the online tutorial by pressing PF1.

For each panel, you can establish local defaults; see [“Defaults for IPCS Dialog Panels” on page 16](#) for more information.

Note: Overriding default values on an IPCS subcommand does not change the SETDEF-defined default. **Use this panel** to make the override a permanent change, or enter the SETDEF subcommand with the requested change.

Example: Specify a default source

The source line contains the default:

```
Source ==> NODSNAME
```

To specify a default source, type the name of the default source over the default:

```
Source ==> DSNAME('IPCSU1.PRBO0075.SVCDUMP')
```

When you press Enter, IPCS changes the SETDEF-defined default source to IPCSU1.PRBO0075.SVCDUMP.

Each of the five default fields may be changed in the same manner.

Use the following commands and PF keys on this panel:

| Command | PF Key |
|-------------------------------|----------|
| CURSOR command (ISPF) | 12 or 24 |
| END primary command (IPCS) | 3 or 15 |
| HELP command (ISPF) | 1 or 13 |
| IPCS command (ISPF) | — |
| RETURN primary command (IPCS) | 2 or 14 |
| SWAP command (ISPF) | 9 or 21 |

Option 1 – BROWSE

The IPCS Browse option displays formatted dump output. Invoke this option by selecting option 1 (BROWSE) from the IPCS Primary Option Menu panel.

```
----- IPCS - ENTRY PANEL -----
COMMAND ==>

CURRENT DEFAULTS:
Source ==> DSNAME('IPCSU1.PR000075.SVCDUMP')
Address space ==> ASID(X'0005')

OVERRIDE DEFAULTS:                                (defaults used for blank fields)
Source ==> DSNAME('D83DUMP.DUMPC.PB00465')
Address space ==> ASID(X'0029')_
Password ==>
POINTER:      Address ==>                          (blank to display pointer stack)
Remark ==>                                          (optional text)
```

Figure 8. IPCS Browse Option Entry Panel

The IPCS Entry panel, [Figure 8 on page 26](#), is the first panel of the IPCS Browse option. Using this panel with pointer and storage panels, the following tasks may be performed:

- Indicate the type of dump source on the entry panel.
- Place pointers associated with addresses of interest into the pointer stack on the pointer panel.
- View storage on a storage panel that is addressed by a requested pointer.
- Format structures in storage referenced by pointers from the pointer panel.
- Enter IPCS subcommands from anywhere in the Browse option.

See “[Browsing a Dump Data Set](#)” on [page 46](#) for information on how to use the entry, pointer, and storage panels of the IPCS Browse option to view a dump.

Option 2 – ANALYSIS

The IPCS MVS Analysis of Dump Contents panel, [Figure 9 on page 26](#), displays a menu of selected high-level dump analysis areas. Invoke this panel by selecting option 2 (ANALYSIS) from the IPCS Primary Option Menu panel.

```
----- IPCS MVS ANALYSIS OF DUMP CONTENTS ----- OPTION ==> _
To display information, specify the corresponding option number.

1SYMPTOMS      - Symptoms                               *****
2STATUS        - System environment summary             * USERID - IPCSU1
3WORKSHEET     - System environment worksheet           * DATE   - 95/06/08
4SUMMARY       - Address spaces and tasks               * JULIAN  - 95.160
5CONTENTION    - Resource contention                   * TIME   - 16:44
6COMPONENT     - MVS component data                    * PREFIX - IPCSU1
7TRACES        - Trace formatting                      * TERMINAL- 3278
                                                    * PF KEYS - 24
                                                    *****

Enter ENDcommand to terminate MVS dump analysis.
```

Figure 9. IPCS MVS Analysis of Dump Contents Panel

Options 1 through 5 provide analysis reports. IPCS processes the requested option for the current default dump source and displays a report that you can view in full-screen mode on the dump display reporter panel.

Option 6 invokes the IPCS Dump Component Data Analysis panel, [Figure 10 on page 28](#), where you can choose a component analysis report. Component analysis is rarely appropriate until the analysis reports provided by options 1 through 5 indicate that a specific component might have incorrect data in its control block(s).

Option 7 invokes the Trace Processing panel, [Figure 11 on page 30](#), where you can choose trace processing reports.

1 SYMPTOMS

Use the SYMPTOMS option to display symptoms collected by SDUMP, recovery routines, and IPCS. The report may provide enough information to match the incident against known problems when you (or the IBM Support Center) does a search of problem data bases. The VERBEXIT SYMPTOMS subcommand generates the same report as this option. See *z/OS MVS Diagnosis: Tools and Service Aids* for symptom examples and *z/OS Problem Management* for information on creating a search argument using these symptoms.

2 STATUS

Use the STATUS option to request a brief description of the system's environment at the time the dump was taken. The STATUS CONTENTION subcommand generates the same report as this option. See the *z/OS MVS IPCS Commands* for sample output.

3 WORKSHEET

Use the WORKSHEET option to display a diagnostic worksheet that describes the state of the system and each processor in the system. The STATUS WORKSHEET subcommand generates the same report as this option. See the *z/OS MVS IPCS Commands* for sample output.

4 SUMMARY

Use the SUMMARY option to request a description of the units of work in current and error address spaces. The SUMMARY subcommand generates the same report as this option. See the *z/OS MVS IPCS Commands* for sample output.

5 CONTENTION

Use the CONTENTION option to request a summary of system-wide resource contention. The ANALYZE subcommand generates the same report as this option. See the *z/OS MVS IPCS Commands* for sample output.

6 COMPONENT

Use the COMPONENT option to display the Dump Component Data Analysis panel as shown in [Figure 10 on page 28](#). On this panel you can select problem analysis reports associated with individual components.

7 TRACE

Use the TRACE option to display the Trace Processing panel as shown in [Figure 11 on page 30](#). On this panel you can select trace processing reports.

Use the following commands and PF keys on this panel:

| Command | PF Key |
|-------------------------------|----------|
| CURSOR command (ISPF) | 12 or 24 |
| END primary command (IPCS) | 3 or 15 |
| HELP command (ISPF) | 1 or 13 |
| IPCS command (ISPF) | — |
| RETURN primary command (IPCS) | 4 or 18 |
| SPLIT command (ISPF) | 2 or 14 |
| SWAP command (ISPF) | 9 or 21 |

After selecting an option, view the report on the dump display reporter panel; see [“Dump Display Reporter Panel” on page 44](#).

Option 2.6 – COMPONENT

The IPCS Dump Component Data Analysis panel, [Figure 10 on page 28](#), displays a menu of diagnostic reports associated with individual components. Invoke this panel by selecting option 6 (COMPONENT) from the Analysis of Dump Contents panel or by entering option 2.6 from the IPCS Primary Option Menu panel.

Use the component analysis panel to request one of these component reports. IPCS processes the request for the current default dump source and displays the requested data on the dump display reporter panel (see [“Dump Display Reporter Panel” on page 44](#)).

```

----- IPCS MVS DUMP COMPONENT DATA ANALYSIS ----- OPTION
===>                                     SCROLL ===> CSR

To display information, specify the corresponding option name or enter S
to the left of the option desired. Enter ? to the left of an option to
display help regarding the component support.

Name      Abstract
ALCWAIT   Allocation wait summary
AOMDATA   AOM Analysis
APPCDATA  APPC analysis
ASCHDATA  APPC transaction scheduler analysis
ASMCHECK  Auxiliary storage paging activity
ASMDATA   ASM control block analysis
AVMDATA   AVM control block analysis
COMCHECK  Operator communications data
COUPLE    XCF Coupling analysis
CTRACE    Component Trace summary
DAEDATA   DAE header data
DB2DATA   DB2 analysis
DIVDATA   Data in virtual storage
DLFDATA   Data Lookaside analysis
DLFTRACE  Data Lookaside trace
GRSDATA   GRS managed resources

```

Figure 10. IPCS MVS Dump Component Data Analysis Panel

Note: Figure 10 on page 28 does not show all the available options. When viewing this panel online, press PF7 and PF8 to scroll through the list. You may not see the options exactly as pictured, depending on the mix of products used at your installation.

Use the following commands and PF keys on this panel:

| Command | PF Key |
|-------------------------------|----------|
| CURSOR command (ISPF) | 12 or 24 |
| END primary command (IPCS) | 3 or 15 |
| HELP command (ISPF) | 1 or 13 |
| IPCS command (ISPF) | — |
| RETURN primary command (IPCS) | 4 or 18 |
| SPLIT command (ISPF) | 2 or 14 |
| SWAP command (ISPF) | 9 or 21 |

After selecting an option, view the report on the dump display reporter panel; see [“Dump Display Reporter Panel” on page 44](#).

Table 2 on page 28 lists the options that are shipped with IPCS and their corresponding subcommands.

| Table 2. Options are shipped with IPCS and corresponding subcommands | |
|---|--|
| Choosing This Component Option: | Produces the Same Report as Does Entering This Subcommand: |
| ALCWAIT — Allocation Wait Summary | VERBEXIT ALCWAIT |
| AOMDATA — Asynchronous Operations Manager Data | VERBEXIT AOMDATA |
| APPCDATA — Advanced Program-to-Program Communications (APPC/MVS) Data | APPCDATA See note at end of table |
| ASCHDATA — APPC/MVS Transaction Scheduler Data | ASCHDATA See note at end of table |
| ASMCHECK — Auxiliary Storage Paging Activity | ASMCHECK |
| ASMDATA — ASM Control Block Analysis | VERBEXIT ASMDATA |

Table 2. Options are shipped with IPCS and corresponding subcommands (continued)

| Choosing This Component Option: | Produces the Same Report as Does Entering This Subcommand: |
|---|--|
| AVMDATA — AVM Control Block Analysis | VERBEXIT AVMDATA |
| COMCHECK — Operator Communications Data | COMCHECK MCSINFO |
| COUPLE — XCF Coupling Data | COUPLE See note at end of table |
| CTRACE — Component Trace Summary | CTRACE QUERY SHORT ALL |
| DAEDATA —DAE Header Data | VERBEXIT DAEDATA |
| DB2DATA —DATABASE 2 | VERBEXIT DSNWDMP |
| DIVDATA —Data-In-Virtual Storage | DIVDATA SUMMARY CURRENT ERROR |
| DLFDATA —Data Lookaside Facility Data | DLFDATA SUMMARY CURRENT |
| DLFTRACE —Data Lookaside Facility Trace | CTRACE COMP(SYSDLF) FULL |
| GRSDATA — GRS managed resources | GRSDATA or VERBEXIT GRSTRACE The GRSDATA panel allows the IPCS user to choose the report type (either GRSDATA or VERBEXIT GRSTRACE). Alternatively, GRSDATA or VERBEXIT GRSTRACE line commands can generate the same report without the need for a panel. |
| IMSDUMP — IMS Analysis | VERBEXIT IMSDUMP |
| IOSCHECK — Active Input/Output Requests | IOSCHECK ACTVUCBS |
| IRLM — IMS Resource Lock Manager | VERBEXIT IRLM |
| JES2 — JES2 Analysis | VERBEXIT HASMFMTM See note at end of table |
| JES3 — JES3 Analysis | VERBEXIT JES3 See note at end of table |
| LISTEDT — Format Eligible Device Table | LISTEDT HEADER |
| LLATRACE — Library Lookaside Trace | CTRACE COMP(SYSLLA) FULL |
| LOGDATA — LOGREC Buffer Record Formatter | VERBEXIT LOGDATA |
| LPAMAP — Map Link Pack Area | LPAMAP ALL |
| MERGE — Merge GTF/CTRACE output | MERGE |
| MMSDATA — MVS Message Service Data | VERBEXIT MMSDATA |
| MTRACE — Master Trace Table Formatter | VERBEXIT MTRACE |
| NUCMAP — Nucleus CSECT Map | VERBEXIT NUCMAP |
| RSMDATA — Real Storage Manager Data | RSMDATA SUMMARY |
| SADMPMSG — Format SADMP Console Messages | VERBEXIT SADMPMSG |
| SMSDATA — Storage Management Subsystem Data | VERBEXIT SMSDATA |
| SRMDATA — SRM Control Block Analysis | VERBEXIT SRMDATA |
| STRDATA — Coupling Facility Structure Data | STRDATA |
| SUMDUMP — Format Summary Dump Data | VERBEXIT SUMDUMP |
| SYMPTOM — Format Symptoms | VERBEXIT SYMPTOMS |
| SYSTRACE — Format System Trace | SYSTRACE |
| TSODATA — TSO/E Analysis | VERBEXIT TSODATA |

Table 2. Options are shipped with IPCS and corresponding subcommands (continued)

| Choosing This Component Option: | Produces the Same Report as Does Entering This Subcommand: |
|---|--|
| VLFDATA — Virtual Lookaside Facility Data | VLFDATA SUMMARY |
| VLFTTRACE — Virtual Lookaside Facility Trace | CTTRACE COMP(SYSVLF) EXCEPTION FULL |
| VSMDATA — VSM Control Block Analysis | VERBEXIT VSMDATA GLOBAL CURRENT ERROR |
| VTAMMAP — VTAM® Control Block Analysis | VERBEXIT VTAMMAP |
| WLMDATA — Workload Manager Data | WLMDATA |
| XESDATA — XES analysis | XESDATA <u>See note below</u> |
| Note: Choosing this component option leads to another panel, which offers all the report options available for the corresponding subcommand. | |

Option 2.7 — TRACE

The IPCS Trace Processing panel, [Figure 11 on page 30](#), displays a menu of trace formatting options. Invoke it by selecting option 7 (TRACE) from the Analysis of Dump Contents panel or by entering option 2.7 from the IPCS Primary Option Menu panel.

```

----- IPCS TRACE PROCESSING -----
OPTION ===>
To display information, specify the corresponding option number.

  1CTTRACE      - Component trace
  2GTFTRACE     - Generalized trace facility
  3MTRACE       - Master trace
  4SYSTEM       - System trace
  5CPUTRACE     - Hardware instruction trace buffer
  6MERGE        - Merge multiple traces
  TTUTORIAL     - Details on these traces

Enter ENDcommand to end IPCS trace processing.
```

Figure 11. IPCS Trace Processing Panel

After choosing a trace processing option (and specifying parameters for certain options), IPCS processes the request for the current default source and displays the formatted trace data on a dump display reporter panel.

• 1 CTRACE

Use the CTRACE option to process component and application traces. This option uses a series of selection panels to provide all possible processing variations of the CTRACE subcommand.

• 2 GTFTRACE

Use the GTFTRACE option to format GTF trace records. This option uses a series of selection panels to provide all possible processing variations of the GTFTRACE subcommand.

• 3 MTRACE

Use the MTRACE option to process the master trace. This option provides the same processing as the VERBEXIT MTRACE subcommand.

• 4 SYSTEM

Use the SYSTEM option to process system trace entries. This option uses a selection panel to provide commonly-used variations of the SYSTRACE subcommand.

• 5 CPUTRACE

Use the CPUTRACE option to process the instruction address trace. This option provides the same processing as the CPUTRACE subcommand.

• 6 MERGE

Use the MERGE option to combine multiple traces into one chronological report. This option uses a series of selection panels to provide all possible processing variations of the MERGE subcommand.

- **T TUTORIAL**

Use the TUTORIAL option to learn more about how to use the trace processing option.

Option 3 – UTILITY

The IPCS Utility Menu panel, [Figure 12 on page 31](#), provides three options for copying data, an option for listing the names of your source data sets, and an option for the dump analysis and elimination (DAE) data set. To invoke it, select option 3 (Utility) from the IPCS Primary Option Menu panel.

```
----- IPCS UTILITY MENU -----
OPTION  ==> _

  1 COPYDDIR - Copy dump directory data
  2 COPYDUMP - Copy a dump data set
  3 COPYTRC  - Copy trace data sets
  4 DSLIST   - Process list of data set names
  5 DAE      - Process DAE data
  6 SADMP    - SADMP dump data set utility

Enter END command to terminate

*****
* USERID - IPCSU1
* DATE   - 95/10/27
* JULIAN  - 95.300
* TIME   - 18:17
* PREFIX  - IPCSU1
* TERMINAL- 3278T
* PF KEYS - 24
*****
```

Figure 12. IPCS Utility Menu Panel

The following describes each utility option:

- **1 COPYDDIR**

Use the COPYDDIR option to copy dump directory data from both one dump directory data set or exported RECFM=VB data set into the current dump directory. This option provides the same function as the COPYDDIR subcommand. The name of dump directory data set or exported data set with one dump description record can be used as a source dsname for this subcommand.

In z/OS Release 2, COPYDDIR is enhanced for sysplex dump directory processing by displaying the information in the directory so that IPCS users can selectively copy it to their own directories.

- **2 COPYDUMP**

Use the COPYDUMP option to copy the contents of an existing dump data set into a data set you specify. This option provides the same function as the COPYDUMP subcommand.

- **3 COPYTRC**

Use the COPYTRC option to copy trace data from one or more dump or trace data sets into a data set you specify. This option provides the same function as the COPYTRC subcommand.

- **4 DSLIST**

Use the DSLIST option to obtain a list of your source data sets. From the list, you can add, browse, delete, edit, or view the data set or enter an IPCS subcommand, CLIST, or REXX exec for the data set.

- **5 DAE**

Use the DAE option to see the symptom strings listed in the DAE data set and take actions on them, if desired.

- **6 SADMP**

Use the SADMP option to perform the tasks associated with creation, clearing, and reallocation of SADMP data sets on DASD.

For more information, press PF1 or type HELP to invoke the online help or reference the corresponding subcommand in the [z/OS MVS IPCS Commands](#).

Option 3.4 – DSLIST

The IPCS Data Set List Utility panel, [Figure 13 on page 32](#), lets you request a list of cataloged data sets. The default for the DSNNAME LEVEL is your TSO/E user ID, even if you fill the field with blanks. In the example screen, it is JOHNDOE. After you press ENTER, the next screen lists all cataloged data sets you can access with the DSNNAME LEVEL value as the first qualifier.

```
-----IPCS DATA SET LIST UTILITY -----
COMMAND ===>

Enter the parameter below:

DSNAME LEVEL ===> JOHNDOE

* The following line commands will be available when the list is displayed.

A - Add data set to IPCS inventory      V - View data set using ISPF
B - Browse data set using ISPF         = - Repeat last command
D - Delete data set                    IPCS subcommand, CLIST or REXX exec
E - Edit data set using ISPF
```

Figure 13. IPCS Data Set List Utility Panel

[Table 3 on page 32](#) shows the line commands for the data set list. You can specify more than one line command; IPCS processes all commands, from the top of the list down, before showing the list again. If processing of a command results in an error code of 8 or greater, IPCS stops processing commands and displays the panel again, with your remaining line commands in place. If you want to continue with the line commands, press ENTER. If not, either:

- Replace the unwanted line commands with blanks and press ENTER.
- Enter an END primary command to leave the data set list.

While entering commands in the data set list, do not use INSERT mode or the DELETE key. Shifting data shown in the list can make IPCS treat all or part of the data set name as part of the command text.

Table 3. Line commands for data set list

| Command | Description |
|---------|---|
| A | <p>Adds the data set to your user dump directory. The A line command is equivalent to an ADDDUMP DSN(/) subcommand. If the ADDDUMP process completes normally, IPCS places the following message next to the data set name:</p> <pre>Added to IPCS inventory</pre> |
| B | <p>Requests browsing of the data set through ISPF browse. If the ISPF browse completes normally, IPCS places the following message next to the data set name:</p> <pre>Viewed using ISPF Browse</pre> |
| D | <p>Requests deletion of either:</p> <ul style="list-style-type: none"> • The source description for the data set in your user dump directory • The data set <p>IPCS displays a panel that allows you to specify what you want deleted. If delete completes normally, IPCS places one of the following messages next to the data set name:</p> <pre>Analysis dropped. Analysis dropped. Data set deleted. All dropped. All dropped. Data set deleted. Translations dropped. Translations dropped. Data set deleted. Data set deleted</pre> |

Table 3. Line commands for data set list (continued)

| Command | Description |
|---------|--|
| E | Requests editing of the data set using ISPF edit. If the ISPF edit completes normally, IPCS places one of the following messages next to the data set name, depending on whether you changed the data set or not: <div> <div>Edited using ISPF Edit</div> <div>Viewed using ISPF Edit</div> </div> |
| V | Requests browsing of the data set through ISPF browse. If the ISPF browse completes normally, IPCS places the following message next to the data set name: <div> <div>Viewed using ISPF Browse</div> </div> |
| = | Requests for this data set the same action taken on the preceding data set. |
| | Enter an IPCS subcommand, CLIST, or REXX exec for the data set. |

Option 3.5 – DAE

The window in Figure 14 on page 33 appears before the DAE Display panel (Figure 15 on page 33).

The first time you invoke the DAE Display panel, the window shows SYS1.DAE as the data set name and no volume serial number. Type the name of the DAE data set over this name, as shown in the example of the window. If you want to view a DAE data set from a system where it is not cataloged, fill in the volume serial (VOLSER).

The system saves your data set name and volume serial, if specified, and displays them the next time the window is shown.

DAE Display Facility Dataset Name Input Panel

DAE Dataset Name . . . 'SYS1.DAE.PLEX'
Volume Serial . . . (Optional)

Figure 14. First window before the DAE Display panel

| ----- DAE Display ----- | | | | | | | | |
|---|----------|--------|--------|-----------|---------|----------|--------------|----------|
| ROW 1 TO 8 OF 305 | | | | | | | | |
| Enter an Action Code next to an entry. Enter / next to an entry to choose from a list of Action Codes. | | | | | | | | |
| Dataset: 'SYS1.DAE.FLEX' | | | | | | | | |
| Dumps since last DAE Display: 2 Total Dumps suppressed: 4623 | | | | | | | | |
| Events since last DAE Display: 15 Suppression rate: 85% | | | | | | | | |
| A | Last | Last | Total | Date of | Symptom | String | information: | |
| C | Date | System | Events | Dump | Abend | Reason | Module | CSECT |
| - | 02/17/25 | MCEVS1 | 318 | 02/17/22 | S0378 | 00000018 | IKFJEFT30 | IKJEFT30 |
| - | 02/14/25 | MCEVS1 | 335 | 02/18/22 | S0878 | 00000010 | IKFJEFT56 | IKJEFT56 |
| - | 02/11/25 | MCEVS1 | 68 | 01/25/23 | S00C4 | 00000011 | IDA0192A | IDA0200T |
| - | 02/07/25 | MCEVS2 | 20 | 02/17/22 | S00F4 | 00000010 | IGWAFMS0 | IGWAMRT0 |
| - | 02/07/25 | MCEVS1 | 2 | 01/30/25 | S00C4 | 00000010 | IDA0192A | IDA0200B |
| - | 02/06/25 | MCEVS1 | 2 | 02/06/25 | S00C4 | 00000011 | IDA0192A | IDA0200B |
| - | 02/06/25 | MCEVS1 | 1 | 02/06/25* | S00C4 | 00000010 | IGDZILLA | IGDERDMP |
| - | 02/04/25 | MCEVS1 | 18 | 02/21/22 | S0301 | | NUCLEUS | IEAVEWAT |

Figure 15. DAE Display Panel

The DAE Display panel, Figure 15 on page 33, shows you the symptom strings in the DAE data set. You can invoke the DAE Display panel in any of the following ways:

- Requesting option 3.5 in the IPCS dialog. If you do not specify the DAE data set name, the system displays an input panel to request the name.

- Entering the TSO/E command at the READY prompt or on the ISPF option 6 panel to request the ADYDSP REXX exec:

```
ADYDSP 'dae-data-set-name'
```

Enter ADYDSP 'dae-data-set-name' DEBUG to see each REXX instruction in ADYDSP and the instruction's variable contents.

- Using the ISPF UTILITY DSLIST option to list data sets you can access. On the ISPF panel for the option, enter:

```
DSNAME LEVEL ==> userid.DAE*
```

On the list of *userid.DAE** data sets, enter ADYDSP next to a DAE data set name.

The volume field is not present if you did not specify a volume serial on the window titled DAE Display Facility Dataset Name Input Panel.

The middle of the panel gives **Total Dumps suppressed** and **Suppression rate**. These calculations assume that MATCH, UPDATE, and SUPPRESS or SUPPRESSALL are in effect. The calculations also assume that one dump was written for each symptom string. The following reasons explain why the values in these fields may not be accurate:

- The ADYSETxx parmlib member being used does not specify the necessary parameters.
- Captured dumps were deleted before being written to DASD. (While captured, dump suppression occurs, but once deleted, the next dump with a matching symptom will be taken.)
- The symptom string contains enough data for a match, but not enough for DAE to determine that it should suppress the dump.

The DAE Display panel shows each symptom string in a line:

- **AC:** The leftmost column, in which you can enter an action code.
- **Last Date:** The last date on which a dump with that symptom string was requested
- **Last System:** The system in the sysplex that requested the last dump.
- **Total Events:** The total number of times the dump has been requested.
- **Date of Dump:** The first date on which the dump was requested. (Note: A trailing asterisk means SYMPTOM STRING TRUNCATED.)
- **Symptom String Information:**
 - **Abend:** The abend code for the dump.
 - **Reason:** The reason code for the dump.
 - **Module:** The module that failed.
 - **CSECT:** The CSECT that failed.

To see more information about a particular symptom string, you can enter an action code in the AC column at the front of that symptom string. [Figure 16 on page 35](#) shows more details about the following symptom string when an action code of "S" is specified:

```
- 01/17/25 MCEVS1          318 02/17/22  S0378 00000018  IKJEFT30  IKJEFT30
```

| ----- DAE Entry Details ----- | | | | ROW 1 TO 8 OF 8 |
|-------------------------------------|---------|------------------------|------------------------------|-----------------|
| Total Events: 318 | | Type: SVC Dump | | |
| Last (most recent) Event: | | Date | Time | System Name |
| Dump Taken: | | 02/17/25 | 12:00:47 | MCEVS1 |
| | | 02/17/22 | 08:38:23 | MCEVS1 |
| Symptoms used for Dump Suppression: | | | | |
| MVS | RETAIN | | | |
| Key | Key | Symptom Data | Explanation | |
| MOD/ | RIDS/ | IKJEFT30 | LOAD MODULE NAME | |
| CSECT/ | RIDS/ | IKJEFT30 | ASSEMBLY MODULE CSECT NAME | |
| PIDS/ | PIDS/ | 566528502 | PRODUCT/COMPONENT IDENTIFIER | |
| AB/S | AB/S | 0378 | ABEND CODE-SYSTEM | |
| REXN/ | RIDS/ | IGX00027 | RECOVERY ROUTINE CSECT NAME | |
| FI/ | VALU/H | 1816100A0D18CE18FB180C | FAILING INSTRUCTION AREA | |
| HRC1/ | PRCS/ | 00000018 | ABEND REASON CODE | |
| SUB1/ | VALUE/C | TSO#I#O#SERVICE#SVC | COMPONENT SUBFUNCTION | |

Figure 16. DAE Display Panel

The DAE Entry Details panel shows more information about the selected symptom string:

- **Last Event:** The last date and time when a dump with that symptom string was requested. The dump may have been suppressed.
- **Dump Taken:** The date and time when the initial dump was produced.
- **Total Events:** The total number of times the dump has been requested. This includes requests that have resulted in dump suppression.
- **Type of Dump:** Either SVC or SYSMDUMP. DAE treats Dumps and transaction Dumps the same. A Transaction Dump will be shown as SYSMDUMP in the panel.

Using the Panel

Use the following to take actions. Details about these actions are in the following topics.

- File, View, and Help on the action bar
- PF keys
- An action code in the leftmost column, with the heading AC
- Primary commands after Command ==>

Action Bar

The actions on the action bar at the top of the panel are File, View, and Help.

The File pull-down menu has options to manage the DAE data set. The options are:

- Refresh the DAE Display panel from the current DAE data set
- List other DAE data sets
- Exit the DAE Display panel.

The View pull-down menu lets you sort the DAE data set data by:

- Last (most recent) date
- Last (most recent) system name
- Total number of events for each symptom string
- Date of dump
- Abend and reason code
- Module name
- CSECT name

The Help pull-down lets you request the following help:

- General ISPF help
- DAE Display panel help

PF Keys

The default PF keys follow. You can tailor your PF keys using ISPF PFKey customization.

PF1 or PF13

Help for the DAE Display panel

PF2 or PF14

Split screen

PF3 or PF15

Exit

PF4 or PF16

Return to the previous panel

PF5

Repeat the previous FIND command

PF6 or PF18

(Not active)

PF7 or PF19

Up

PF8 or PF20

Down

PF9 or PF21

Swap screens

PF10 or PF22

Move the cursor to the action bar

PF11 or PF23

(Not active)

PF12

End processing with this panel

PF17

Keylist

PF24

Retrieve

Action Codes

The action codes entered in the leftmost AC column are:

S

Show all the symptoms for this record in the DAE data set. For special conditions, such as, SYMPTOM STRING TRUNCATED, the panel may display a message.

T

Take (the TAKEDUMP option) the next dump for this symptom string, then resume suppressing subsequent dumps for this symptom string.

Note:

1. You (through the invoking user ID) must have WRITE access to the DAE data set.
2. You have set up the system and the invoking userid correctly. See the explanation on generating a suppressed dump in [z/OS MVS Diagnosis: Tools and Service Aids](#).
3. If a CHNGDUMP or SLIP operator command suppresses the dump, the T action code will not generate the dump.

4. The commands necessary to accomplish the task are issued by TSO users at their dispatching priority. It is possible that the system may not dispatch the TSO user due to that dispatching priority, and therefore the action may not complete in a timely manner.

V

View the source description for the dump with this symptom string in your current dump directory. IPCS displays the title on the IPCS INVENTORY panel (IPCS Option 4). If IPCS cannot find a dump with the symptom string in your directory, the panel displays a message.

/ (slash)

Show a window with action code choices.

Primary Commands

You can enter the following primary commands:

EXIT

Leaves the DAE Display panel. This command is the same as PF3 or as EXIT in the pull-down menu for File on the action bar.

FIND *string* or F *string*

Searches for the specified string beginning at the top of the screen. The string cannot contain embedded blanks.

FIND *string* PREV

Searches for the specified string beginning on the line above the cursor.

HELP

Displays help for the DAE Display panel. This command is the same as option 2 in the pull-down menu for Help on the action bar.

LISTD

Lists other DAE data sets with a default pattern of SYS1.*DAE*. You can also specify your own pattern, for example:

```
LISTD SYS2.DAE*
```

This command is the same as the LISTD option in the pull-down menu for File on the action bar.

REFR

Refreshes the DAE Display from the current DAE data set. This command is the same as the REFR option in the pull-down menu for File on the action bar.

RFIND

Resumes the search beginning on the next line, if the cursor has not moved, or on this line, if the cursor has moved. This command is the same as PF5.

SORT *heading* or SO *heading* or SOR *heading*

Sorts the symptom strings by a column. The *heading* is the last word in the heading for the column.

- date
- system
- events
- dump
- abend
- reason
- module
- csect

The abend and reason fields are treated as a single sort field.

This command is the same as:

- Placing the cursor on the last line of a column heading and pressing ENTER

- Requesting a sort using the pull-down menu for View on the action bar

Option 3.6 – SADMP

Select option 3.6 to use the SADMP DASD dump data set panel to create, clear, and reallocate SADMP data sets on DASD.

```
----- SADMP DASD Dump Data Set Utility -----
Command ==> -----

Enter/verify parameters.
Use ENTER to perform function, END to terminate.

Function ==> R (C - Clear, D - Define, R - Reallocate)
DSNAME ==> MY.SADMP-----

Volume serial numbers: (1-32)
  1- 8 VOL001 -----
  9-16 -----
 17-24 -----
 25-32 -----

Unit ==> 3390 (3380, 3390, or 9345)
Cylinders ==> 500 (cylinders per volume)

DSNTYPE ==> B (B - Basic, L - Large, E - ExtReq)
CATALOG ==> Y (Y or N)
EATTR ==> N (N - No, O - Optional)

Optional SMS classes: (May be required by installation ACS routines)
StorClas ==> ----- DataClas ==> ----- MgmtClas ==> -----
```

Figure 17. SADMP DASD dump data set panel

This utility performs the same functions associated with the AMDSADDD REXX utility. You can also use AMDSADDD, but references to SAMPLIB must now refer to ABLSCLIO. The data set is placed in SBLSCIO rather than SAMPLIB because it is no longer a sample. For more information on how to use the AMDSADDD Utility, see [Using the AMDSADDD Utility in z/OS MVS Diagnosis: Tools and Service Aids](#).

Tip: Systems and the applications that they support tend to get larger and more complex over time. This impacts the dumps and traces that they produce and, in turn, may create problems for you when you attempt to analyze problems using IPCS. The tactics described in this chapter attempt to guide you to use IPCS most effectively.

Option 4 – INVENTORY

The IPCS Inventory panel, [Figure 18 on page 38](#), helps you review and manage the sources described by a dump directory. Invoke this panel by selecting option 4 (INVENTORY) from the IPCS Primary Option Menu panel.

```
IPCS Inventory - 'USER1.DDIR' -----
COMMAND ==> ----- SCROLL ==> CSR

AC Dump Source                                     Status
-----
-- ACTIVE                                           CLOSED
No title
No symptoms
-- DDNAME(ISPROF)                                   CLOSED
No title
No symptoms
-- DSNAME('D83DUMP.SYS1.INF0025.IOSVSLFD')          ADDED
Title=COMPON=IOS-SELF DESCRIPTION SERVICE,COMPID=SC1C3,ISSUER=IOSVSLFD
Psym=RIDS/NUCLEUS#L RIDS/IOSVSLFD PIDS/5752SC1C3 AB/S00C1 RIDS/IOSVSLFD#R REGS/
-- DSNAME('USER1.GTF.TRACE')                        CLOSED
Title=GTF trace
No symptoms - not an MVS/SP unformatted dump
-- DSNAME('USER1.SYS3.DUMP.S0C4')                    CLOSED
Title=JOBNAME USER1 STEPNAME SMPROC SMPROC SYSTEM 0C4
Psym=RIDS/BLSG#L AB/S00C4 VALU/HD08C4170 REGS/0C374 PRCS/00000004
***** END OF IPCS INVENTORY *****
```

Figure 18. IPCS Inventory Panel

All IPCS users can review and manage their user dump directory with this panel. If you are authorized, you can use this panel to review and manage the sysplex dump directory.

The status of a source can be:

- **ADDED:** The source description was added by ADDDUMP processing to your user dump directory. This status indicates that IPCS must initialize the source before you can format it.
- **CLOSED:** The source is not currently being used.
- **OPEN:** The source is currently being used.

The first line under the data set ddname or name contains the dump title, or an indication that there is no dump title. The second line contains, if available, as much of the symptom string as will fit on the line. The type of symptom string is indicated to the left of the string, as shown in [Table 4 on page 39](#).

| Table 4. Types of symptom strings | | |
|-----------------------------------|------------------------|---|
| Type | Symbol in Symbol Table | Source of the Symptom String |
| Plex® | REMOTEDUMP | If available, the information about the remote dump and the associated system name. |
| Trap | SLIPTRAP | If available, the text of the trap that requested the dump. |
| Psym | PRIMARYSYMPTOMS | The symptoms provided by the program that requested the SVC dump or SYSMDUMP dump. |
| Ssym | SECONDARYSYMPTOMS | The symptoms collected by IPCS while processing an SVC dump requested by a DUMP operator command or a stand-alone dump. |

To use this panel, enter a 2-character command code in the AC column. These command codes let you request actions directly from this panel. For more information, see [z/OS MVS IPCS Commands](#).

BR

storage

CL

Close the source

DD

Delete the source description

DT

Delete the translation records from the source description

LA

List the source description with storage attributes

LB

List the source description with record locations

LD

List the source description with a dumped storage summary

LT

List the source description with translation results

LZ

List the source description with information from all list codes

OP

Open a source

SD

Establish this source as the current source in your local defaults

XP

Export dump description to RECFM = VB data set

You can scroll through the sources using the UP, DOWN, and LOCATE primary commands. You can also use PLEX primary command to display only the dumps related to multi-system incident.

Option 5 – SUBMIT

The IPCS MVS Dump Batch Job Option Menu panel offers batch processing of dumps or other problem data, including logrec records. Invoke this panel by selecting option 5 (SUBMIT) from the IPCS Primary Option Menu panel.

```
----- IPCS MVS DUMP BATCH JOB OPTION MENU -----
OPTION ==> _

      1SADUMP   - Prepare stand alone dump for analysis
      2SVCDUMP  - Prepare SVC dump for analysis
      3SYSMDUMP - Prepare SYSMDUMP for analysis
      4SUPPLEMENT - Perform supplementary dump analysis
      5EREP     - Process software data using EREP
      6DPfD     - Data Privacy for Diagnostics
JOB STATEMENT INFORMATION: (Verify before proceeding)

      ==> //USER1 JOB ACCT57,'IBM PSR',NOTIFY=USER1,
      ==>

*****
* USERID - USER1
* DATE   - 95/05/17
* JULIAN  - 95.137
* TIME   - 16:45
* PREFIX  - USER1
* TERMINAL - 3278
* PF KEYS - 24
*****
      ==> // MSGCLASS=A,MSGLEVEL=(2,1)      ==>      ==>

Enter END to end batch job processing.
```

Figure 19. IPCS MVS Dump Batch Job Option Menu Panel

Use this panel to select the type of batch job and to set up the JOB statement to be used with the job.

Note:

1. When you submit a batch job from any one of these panels, IPCS requires exclusive use of your user dump directory. Either submit the batch job using a different dump directory than your current directory or free your current user dump directory for the batch job to run.
2. The job submission dialog assumes that the SYS1.CMDLIB system data set is in the system LNKLIST concatenation.

For quick turn-around, run your job with the output directed to a hold queue. When the job completes, view the output online by:

- Using a TSO/E OUTPUT command to copy the output into a TSO/E data set
- Using the PDF BROWSE option

• 1 SADUMP

Use the SADUMP option to prepare a stand-alone dump that is on **tape** for dump analysis. It submits a job that copies the dump to DASD and invokes the BLSCSCAN CLIST to generate an initial dump analysis screening report.

Note: If the stand-alone dump is already on DASD, use SUPPLEMENT.

• 2 SVCDUMP

Use the SVCDUMP option to prepare an SVC dump for dump analysis. It submits a job that invokes the BLSCBSVB CLIST to generate a diagnostic screening report.

• 3 SYSMDUMP

Use the SYSMDUMP option to prepare a SYSMDUMP dump for problem analysis. It submits a job that invokes the BLSCBSYB CLIST to generate a diagnostic screening report.

• 4 SUPPLEMENT

Use the SUPPLEMENT option to enter an IPCS subcommand, CLIST, or REXX exec to be processed as a batch job against a designated dump.

To use this option to prepare a stand-alone dump that is on **DASD** for dump analysis, specify BLSCSCAN as the CLIST to be invoked.

• 5 EREP

Use the EREP option to request the Environmental Record Editing and Printing Program (EREP) to process logrec records. EREP places the results in a SYSOUT data set, which you can read online.

After you provide the JOB statement and select option 5, IPCS displays the Process Software Data Using EREP panel. On this panel you select:

- The type of the input and, for a history data set or logrec data set, the name of the data set. For input from the log stream, IPCS ignores the data set name field, if filled in.

If you enter the full data set name, including the first qualifier, enclose the data set name in apostrophes. If you omit the first qualifier, do not enclose the name in apostrophes; the system will add a first qualifier, which is the data set name prefix in your TSO/E profile.

- The SYSOUT class for the output and, if desired, other JCL options for SYSOUT. IPCS does not check the other options. If you select a held queue for the output, you can see it online as soon as the job completes.
- Other options that limit the number of logrec records to be formatted. Considerations for these options are:

- Start and stop times must be in Greenwich Mean Time (GMT) for records from the logrec data set or a history data set. The stop time should include the last minute. For example, to process records for an entire day, specify a start time of 00:00 and a stop time of 24:00. If the stop time is 23:59, the last minute's records are not included.

To make sure that all records are included, IPCS adds 24 hours to the specified stop time. The reason is that the logrec records are stored in a single block that is time stamped after the last record is stored; this time stamp may be much later than the time of the desired records.

- In the OTHER OPTIONS field, IPCS initially supplies TYPE=S, meaning only software logrec records.

• 6 Data Privacy for Diagnostics

Use this option to post-process SVC, stand-alone, or SLIP dumps taken on z15™ or later processors in order to redact pages that have been tagged as being sensitive by the applications that created the pages, as well as untagged pages that will be scanned and detected as containing sensitive data per the Data Privacy for Diagnostics Analyzer. For additional information, see the topic Using Data Privacy for Diagnostics in the z/OS MVS IPCS User's Guide.

The IPCS dialog provides online help for specifying the parameters of each job. Press PF1 or type HELP to access the online help.

See [z/OS MVS IPCS Commands](#) for more information about the CLISTs that this option invokes.

Option 6 – COMMAND

The IPCS Subcommand Entry panel, [Figure 20 on page 41](#), provides a means for entering IPCS subcommands and invoking CLISTs. Invoke this panel by selecting option 4 (COMMAND) from the IPCS Primary Option Menu panel.

```
----- IPCS Subcommand Entry -----
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation below:

===> _

----- IPCS Subcommands and Abbreviations -----
ADDUMP      DROPDUMP, DROPD      LISTMAP, LMAP      RUNCHAIN, RUNC
ANALYZE      DROPMAP, DROPM      LISTSYM, LSYM      SCAN
ARCHECK      DROPSYM, DROPS      LISTUCB, LISTU     SELECT
ASCBEXIT, ASCBX EQUATE, EQU, ED  LITERAL            SETDEF, SETD
ASMCHECK, ASMK  FIND, F          LPAMAP             STACK
CBFORMAT, CBF  FINDMOD, FMOD      MERGE              STATUS, ST
CBSTAT        FINDUCB, FINDU      NAME                SUMMARY, SUMM
CLOSE         GTFTRACE, GTF       NAMETOKN            SYSTRACE
COPYDDIR      INTEGER            NOTE, N              TCBEXIT, TCBX
COPYDUMP      IPCS HELP, H        OPEN                VERBEXIT, VERBX
COPYTRC       LIST, L             PROFILE, PROF        WHERE, W
CTRACE        LISTDUMP, LDMP      RENUM, REN
```

Figure 20. IPCS Subcommand Entry Panel

Use this panel to enter IPCS subcommands and invoke CLISTs and REXX execs as you would in IPCS line mode. IPCS displays the output from the requested subcommand, CLIST, or REXX exec in full-screen

mode on the dump display reporter panel. (See [“Dump Display Reporter Panel”](#) on page 44, for more information). For your reference, this panel lists all the IPCS subcommands and abbreviations below the entry line.

Note:

1. If the Subcommand Entry panel cannot process the request, IPCS generates a return code of 16. Otherwise, the return code from the subcommand, CLIST or REXX exec is returned.
2. To enter the HELP subcommand from this panel, use only the abbreviated form of this subcommand: **H**.
3. See the TSO subcommand for information about entering CLISTs and REXX execs.

Use the following commands and PF keys on this panel.

| Command | PF Key |
|-------------------------------|----------|
| CURSOR command (ISPF) | 12 or 24 |
| END primary command (IPCS) | 3 or 15 |
| HELP command (ISPF) | 1 or 13 |
| IPCS command (ISPF) | — |
| RETURN primary command (IPCS) | 4 or 18 |
| SPLIT command (ISPF) | 2 or 14 |
| SWAP command (ISPF) | 9 or 21 |

After entering a subcommand, CLIST, or REXX exec, view the report on the dump display reporter panel; see [“Dump Display Reporter Panel”](#) on page 44.

Option T — TUTORIAL

The IPCS Dialog Tutorial option provides immediate, online reference and instruction on how to use the IPCS dialog. Invoke this option by selecting option T (TUTORIAL) from the IPCS Primary Option Menu panel. The first three panels explain how the tutorial is organized and how you can use it. [Figure 21](#) on page 42 shows the first of these three panels. You can scan the tutorial sequentially from beginning to end, or you can select specific topics from a table of contents. It is quite similar to the ISPF tutorial.

```
----- IPCS TUTORIAL -----
COMMAND ==> _

                IPCS Problem Analysis Dialog
                TUTORIAL

This tutorial provides online information about the features and operation
of the Interactive Problem Control System problem analysis dialog (IPCS).
You may view the tutorial sequentially, or you may choose selected topics
from lists that are displayed on many of the tutorial pages.

The table of contents contains a list of major topics. Subsequent pages
contain additional lists that lead you to more specific levels of detail.

The next two pages contain a description of how to use this tutorial.

Press ENTER key to proceed to the next page, or
Enter UP command to go directly to the table of contents, or
Enter END command to return to the Primary Option Menu.
```

Figure 21. IPCS Dialog Tutorial Panel

When using the tutorial, the IPCS primary and line commands to scroll through a screen are interpreted as follows:

UP/PF7

Display a higher-level list of topics.

DOWN/PF8

Skip to the next topic.

LEFT/PF10

Display the previous tutorial page.

RIGHT/PF11

Display the next tutorial page.

Note: Pressing Enter also causes the next tutorial panel to be displayed.

To display a one-page summary of how to use the tutorial, at any time, enter HELP on the command line of the tutorial.

IPCS Dialog Panels

The IPCS dialog consists of a series of panels. These panels may be grouped together according to the function they provide in the IPCS dialog. For example:

- Selection panels, such as the IPCS Primary Option Menu panel, offer a menu of options.
- Data entry panels, such as the IPCS Default Values panel, provide fields to enter data such as job parameters, data set names, etc.
- The pointer and storage panels of the IPCS Browse option organize the process of viewing formatted dump data.
- Dump display reporter panels display dump analysis and formatting reports.

Depending on the type of panel, the acceptable commands and PF key settings differ.

z/OS MVS IPCS Commands lists the valid commands and PF keys for each panel type and describes the syntax and parameters for the IPCS primary and line commands used on the panels.

Selection and Data Entry Panels

Figure 9 on page 26 is an example of a selection panel. On this panel select from a list of options by entering its number on the COMMAND/OPTION line. Figure 7 on page 25 is an example of a data entry panel. On this panel, supply parameters by filling in labeled fields. Many fields retain previous values.

Pointer and Storage Panels

The IPCS Browse option has three kinds of panels, which are explained in [“Browsing a Dump Data Set”](#) on page 46.

- The data entry panel, which is explained in the previous topic
- The pointer panel
- The storage panel

Use the following commands and PF keys on these panels:

| Command | PF Key |
|---------------------------------|----------|
| CANCEL primary command (IPCS) | — |
| CBFORMAT primary command (IPCS) | — |
| CURSOR command (ISPF) | 12 or 24 |
| D (delete) line command (IPCS) | — |
| DOWN primary command (IPCS) | 8 or 20 |
| E (edit) line command (IPCS) | — |
| END primary command (IPCS) | 3 or 15 |
| EQUATE primary command (IPCS) | — |

| Command | PF Key |
|---------------------------------------|---------|
| F (format) line command (IPCS) | — |
| FIND primary command (IPCS) | — |
| HELP command (ISPF) | 1 or 13 |
| I (insert) line command (IPCS) | — |
| IPCS primary command (IPCS) | — |
| LOCATE primary command (IPCS) | — |
| LOCATE CURSOR% primary command (IPCS) | — |
| LOCATE CURSOR? primary command (IPCS) | — |
| R (repeat) line command (IPCS) | — |
| RENUM primary command (IPCS) | — |
| RESET primary command (IPCS) | — |
| RETURN primary command (IPCS) | 4 or 16 |
| RFIND primary command (IPCS) | 5 or 17 |
| S (select) line command (IPCS) | — |
| SELECT primary command (IPCS) | — |
| SPLIT command (ISPF) | 2 or 14 |
| STACK primary command (IPCS) | 6 or 18 |
| STACK X primary command (IPCS) | — |
| SWAP primary command (ISPF) | 9 or 21 |
| UP primary command (IPCS) | 7 or 19 |
| WHERE primary command (IPCS) | — |

For line commands and selection codes valid on these panels, see [“Working from the Pointer Stack”](#) on page 48 and [“Selecting Words from the Storage Panel”](#) on page 51.

Dump Display Reporter Panel

When an IPCS subcommand, CLIST, or REXX exec is entered on the IPCS Subcommand Entry panel, [Figure 20 on page 41](#), you can request that the dump data output be transmitted to either the terminal or the IPCS print data set by using the message routing parameters. (If a message routing parameter is not specified, the default for this parameter is used.)

When you direct the output to the IPCS print data set, you can view the dump data by using the ISPF/PDF Browse option. When you direct the output to the terminal, you can view the dump data in full-screen mode on the dump display reporter panel.

```

IPCS OUTPUT STREAM ----- LINE 0 COLS 1 78
COMMAND ==> _ SCROLL ==> CSR
***** TOP OF DATA *****
* * * * K E Y F I E L D S * * * *
JOBNAME IPCSU1
SELECTED BY: CURRENT
ASCB: 00920200
FWDP..... 00914E00 ASID..... 00B3 CSCB..... 025D34D0 DSP1..... 00
TSB..... 00922178 AFFN..... FFFF ASXB..... 005FDC20
FLG2..... 00 SRBS..... 0000 LOCK..... 00000000
ASSB..... 01929980
TCB: 005FDE40
CMP..... 00000000 PKF..... 00 LMP..... FF DSP..... FF
TSFLG.... 00 STAB..... 005FDDF8 NDSP..... 00000000
JSCB..... 005FDAA4 BITS..... 00000000 DAR..... 00
RTWA..... 00000000 FBYT1.... 00 STCB..... 7FFFECEB0
PRB: 005FDAD8
WLIC..... 00020001 FLCDE.... 00BF9458 OPSW..... 070C1000 810203A0
LINK..... 015FDE40
CDE: 00BF9458

```

Figure 22. Dump Display Reporter Panel — Result of Processing a Subcommand

Figure 22 on page 45 shows the dump display reporter panel with the output from the IPCS SUMMARY subcommand. Reports generated from the ANALYSIS and COMMAND options of the IPCS Primary Option Menu and from the format line command in the Browse pointer panel appear in this panel.

Use the following commands and PF keys on this panel:

| Command | PF Key |
|---------------------------------|----------|
| CBFORMAT primary command (IPCS) | — |
| CURSOR command (ISPF) | 12 or 24 |
| D (delete) line command (IPCS) | — |
| DOWN primary command (IPCS) | 8 or 20 |
| END primary command (IPCS) | 3 or 15 |
| F (show) line command (IPCS) | — |
| FIND primary command (IPCS) | — |
| HELP command (ISPF) | 1 or 13 |
| IPCS primary command (IPCS) | — |
| L (show) line command (IPCS) | — |
| LEFT primary command (IPCS) | 10 or 22 |
| LOCATE primary command (IPCS) | — |
| MORE primary command (IPCS) | 6 or 18 |
| RESET primary command (IPCS) | — |
| RETURN primary command (IPCS) | 4 or 16 |
| RFIND primary command (IPCS) | 5 or 17 |
| RIGHT primary command (IPCS) | 11 or 23 |
| S (show) line command (IPCS) | — |
| SPLIT command (ISPF) | 2 or 14 |
| SWAP primary command (ISPF) | 9 or 21 |
| UP primary command (IPCS) | 7 or 19 |
| WHERE primary command (IPCS) | — |
| X (exclude) line command (IPCS) | — |

Browsing a Dump Data Set

The Browse option of the IPCS dialog consists of a series of three panels that provide a formatted view of dump data. The entry panel allows you to choose which dump you would like to view. The pointer panel offers quick, organized access to points of interest in the dump. The storage panel provides the view of the particular dump at a specified address.

z/OS MVS IPCS Commands contains a list of the valid commands and PF keys for these panels.

Entry Panel

```
----- IPCS - ENTRY PANEL -----  
COMMAND ==>  
  
CURRENT DEFAULTS:  
Source ==> DSNAME('IPCSU1.PR00075.SVCDUMP')  
Address space ==> ASID(X'0005')  
  
OVERRIDE DEFAULTS: (defaults used for blank fields)  
Source ==> DSNAME('D83DUMP.DUMPC.PB00465')  
Address space ==> ASID(X'0029')  
Password ==>  
POINTER:  
Address ==> (blank to display pointer stack)  
Remark ==> (optional text)
```

Figure 23. Overriding Defaults on the Browse Option Entry Panel

Figure 23 on page 46 shows the first panel displayed after choosing option 1 (BROWSE) on the IPCS Primary Option Menu panel. On this panel, specify the dump source to be processed and the address space within that dump source.

The source and address space fields under CURRENT DEFAULTS will display the SETDEF-defined default dump source and address space. You can override these defaults for a browse session by specifying a new source and address space under OVERRIDE DEFAULTS.

Specifying a Dump Source

Specify any one of the following as an override source:

- System dumps
- System storage
- Partitioned data set (PDS) directories
- Members of PDSs
- Sequential data sets
- VSAM objects

Figure 23 on page 46 shows that the current default dump source is data set IPCSU1.PR00075.SVCDUMP. This dump source, however, will not be processed. Instead, dump data set D83DUMP.DUMPC.PB00465 will be processed because the user, in this example, supplied an overriding dump source.

Once an overriding dump source is entered, IPCS retains that dump source until the overriding dump source is changed on this panel or the Browse option is exited or ended. The override defaults do not change the SETDEF-defined defaults.

Note: Because the override defaults do not change the SETDEF-defined current defaults, subcommands entered in the Browse option through the IPCS primary command will display information about the **current** default source and not about the override default source.

Specifying an Address Space

If an address space is entered, IPCS accepts the address processing parameters, which are used to define an address space. IPCS supplies a default address space as follows if no address space is designated for a dump source:

- IPCS uses the ASID recommended by the dumping program for MVS virtual dump source.
- For stand-alone dumps, IPCS uses ASID(X'0001'). IPCS obtains the CPU parameter value by determining if a valid store-status operation was done. If yes, the IPLed CPU address is used; otherwise, another processor's address in the configuration is used. If no CPU address can be obtained, IPCS uses NOCPU.
- IPCS uses the RBA address space for all other dump sources.

The address space is used to qualify literal and indirect addresses until it is overridden. This can be accomplished by:

- Returning to this panel and entering the address space identifier (ASID).
- Editing the address space associated with a pointer on the pointer panel.
- Using the LOCATE primary command designating an address within another space.

Specifying the Address of Storage

If an address, such as 00FCE1F8, is entered in the address field under POINTER, the next panel to be displayed is the storage panel. (See [Figure 29 on page 50.](#)) This panel shows the dumped storage beginning at the requested address.

If no address is entered, the next panel to be displayed is the pointer panel. See [Figure 24 on page 47.](#)

Pointer Panel

The pointer panel of the Browse option displays a set of pointers for a source. A pointer can take you directly to a position of interest in a dump.

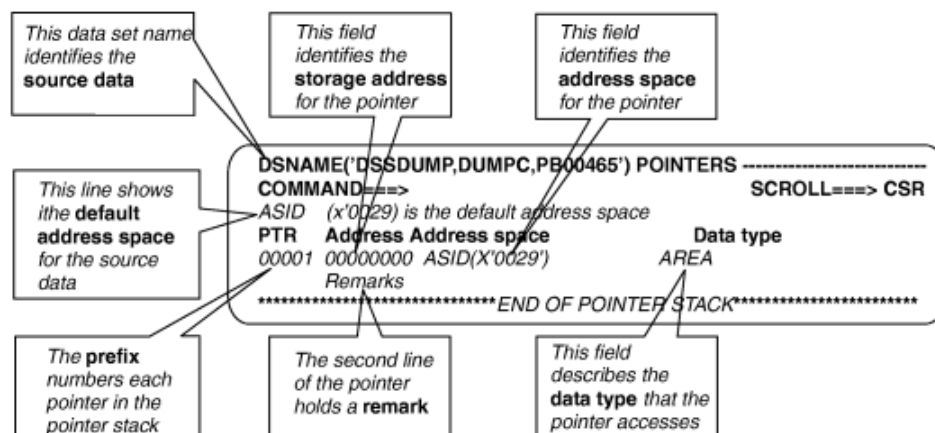


Figure 24. IPCS Dialog Pointer Panel

Figure 24 on page 47 shows the initial display of the pointer panel after you select a dump source and press Enter on the preceding entry panel. (See [Figure 23 on page 46.](#)) A pointer consists of an address, an address space identifier, and a brief remark pertaining to its position. The pointer stack is retained after you end your IPCS session.

You can add pointers to the pointer stack. See [“Selecting Words from the Storage Panel” on page 51](#) and the STACK subcommand for more information.

Working from the Pointer Stack

All the pointer fields in the stack can be edited or displayed. You select a pointer for processing by entering a 1-character line command in the field at the left of the pointer panel display. Use the following line commands; for more information about the commands see [z/OS MVS IPCS Commands](#).

D

Delete the pointer from the stack

E

Edit the pointer the address, address space, data type, and remark

F

Format and display the structure described by the pointer

I

Insert a pointer after this pointer

R

Repeat this pointer immediately following this pointer

S

Select the storage addressed by the pointer for display (see [“Displaying the Dump Data”](#) on page 49)

You can select multiple pointers in a single transaction; they are processed in order of appearance on the screen.

If any of the entries in the scrollable area of the pointer panel are not valid, IPCS highlights the error field and displays a message in the upper-right corner of the display panel. After you correct the error, IPCS displays the pointer table again (unless you enter CANCEL).

Note: Creating a new pointer also adds a new symbol to the symbol table. See [“Adding and Deleting Symbols from the Pointer Stack”](#) on page 93 for more information.

Example: Using line commands

[Figure 25 on page 48](#) shows the I and R line commands entered in the pointer stack.

```
DSNAME('D83DUMP.DUMPC.PB00465') POINTERS -----
COMMAND ==>                                SCROLL ==> CSR
PTR  Address  Address space  Data type
00001 00000000 HEADER      AREA
      Remarks: COMMENT 1
i0002 00FCE210 ASID(X'0029') AREA
      Remarks: COMMENT 2
00003 00000DB0 ASID(X'0008') AREA
      Remarks:
r2004 00F9AC68 ASID(X'0029') AREA
      Remarks:
00005 00FCE1F8 ASID(X'0001') STRUCTURE(CVT)
      Remarks: COMMUNICATIONS VECTOR TABLE
***** BOTTOM OF DATA *****
```

Figure 25. Entering Line Commands in the Pointer Stack of the Browse Option Pointer Panel

[Figure 26 on page 49](#) shows the result of processing the I and R line commands. Processing of I created a new pointer (00003). The new pointer does not supply an address (00000000); it is assigned to ASID(X'0029'). Processing of R followed by the number 2 created two new pointers (00006 and 00007), each of which is identical to the original pointer. Note that inserting and replicating these pointers renumbered the pointer stack.

```

DSNAME('D83DUMP.DUMPC.PB00465') POINTERS -----
COMMAND ==>
PTR Address Address space Data type SCROLL ==> CSR
00001 00000000 HEADER AREA
Remarks: COMMENT 1
00002 00FCE210 ASID(X'0029') AREA
Remarks: COMMENT 2
00003 00000000 ASID(X'0029') AREA
Remarks: COMMENT 3
00004 00000DB0 ASID(X'0008') AREA
Remarks:
00005 00F9AC68 ASID(X'0029') AREA
Remarks:
00006 00F9AC68 ASID(X'0029') AREA
Remarks:
00007 00F9AC68 ASID(X'0029') AREA
Remarks:
00008 00FCE1F8 ASID(X'0001') STRUCTURE (CVT)
Remarks: COMMUNICATIONS VECTOR TABLE
***** BOTTOM OF DATA *****

```

Figure 26. Storage Panel Displayed after Entering Line Commands in the Pointer Stack of the Browse Option Pointer Panel

Displaying the Dump Data

From the pointer panel, you can select a pointer that displays dump data on a storage panel by either entering the SELECT primary command with the requested pointer number on the command line (line two of the display) or by entering the SELECT command directly in the pointer stack.

Figure 27 on page 49 repeats the same display as shown in Figure 24 on page 47 except that an S (a selection code) has been entered in the pointer stack and the word HEADER has been entered for the address space.

```

DSNAME('D83DUMP.DUMPC.PB00465') POINTERS -----
COMMAND ==>
ASID (X'0029') is the default address space
PTR Address Address space Data type SCROLL ==> CSR
s0001 00000000 (HEADER) AREA
Remarks
***** BOTTOM OF DATA *****

```

Figure 27. Selecting a Pointer on the Browse Option Pointer Panel

When you press Enter on the preceding pointer panel, IPCS displays the dump data on the storage panel, beginning with address 00000000 for the header record. (See [Figure 28 on page 50](#).) The dump header record provides you with diagnostic information such as:

- The program that requested the dump: IKJDAIR.
- The title of the dump: TSO OUTPUT CP ESTAE.
- The current address space that was running when the dump was generated. The ASID appears in the last two bytes of the first word of the referenced storage, X'FEFF001D'.

```

HEADER STORAGE-----
COMMAND ==>
00000000. FEF0001D 40000000 C9D2D1C4 C1C9D940 | ....IKJDAIR |
00000010. 931C47FD ED0F9A40 03000098 30810000 | 1.....q.a.. |
00000020. E3E2D640 D6E4E3D7 E4E340C3 D740C5E2 | TSO OUTPUT CP ES |
00000030. E3C1C540 40404040 40404040 40404040 | TAE |
00000040. 40404040 40404040 40404040 40404040 | |
00000050. 40404040 40404040 40404040 40404040 | |
00000060. 40404040 40404040 40404040 40404040 | |
00000070. 40404040 40404040 40404040 40404040 | |
00000080. 40404040 00000000 00000000 00000000 | ..... |
00000090. 00000000 00000000 00000000 00000000 | ..... |
000000A0. 00000000 00000000 00000005 000BDE10 | ..... |
000000B0. 000BDBEC 0000003B 00000000 FFFFB78 | ..... |
000000C0. 000BDD8 007C4B70 007C8658 007F1790 | ...Q.@...@f... |
000000D0. 00006FE8 000BDD14 400B8F58 000BDE88 | ..?Y...."....h |
000000E0. 700B91BA 00000000 5EB0EE40 01D3D07F | ..j.....;".L " |
000000F0. 00000000 C000001D 0000001D 81C17000 | .....aA.. |
00000100. FE000000 01D3D07F 00000000 00000000 | .....L "..... |
00000110. 00000000 00000000 01C3D96F 00000000 | .....CR?.... |
00000120. DF881C02 00000000 070C3000 000B9204 | .h.....k. |
00000130. 01400060 00000000 00000000 01E80348 | .-.....Y.. |
00000140. 00FCE210 000010DA 0042001D 00087B69 | ..S.....#. |

```

Figure 28. Result of Selecting a Pointer on the Browse Option Pointer Panel

Specifying an Address on the Pointer Panel

IPCS allows the address of the pointer to be either a literal or a symbolic address. Because of this, literal addresses that begin with a leading letter must end with a period for IPCS to treat the address as a literal address. Otherwise IPCS will interpret the address as a symbol.

Storage Panel

Through both the entry panel and the pointer panel of the Browse option, you can request to display source data. (See “Displaying the Dump Data” on page 49 and “Specifying the Address of Storage” on page 47.) In response to a storage display request, IPCS displays a storage panel similar to the one shown in Figure 29 on page 50.

The diagram shows the IPCS Storage Panel with the following callouts:

- The top left corner identifies the address space**: Points to the header 'ASID(X'0029') STORAGE'.
- The first column gives the storage address**: Points to the first column of hexadecimal values.
- The next four columns contain a hexadecimal representation of storage**: Points to the next four columns of hexadecimal values.
- Enter selection codes in the space to the left of each hexadecimal column**: Points to the space between the first and second columns.
- This area contains an EBCDIC representation of storage**: Points to the rightmost column containing EBCDIC characters.

The panel content is as follows:

```

ASID(X'0029') STORAGE-----
COMMAND ==>
00000000 FEF0001D 40000000 C9D2D1C4 C1C9D940 | ....IKJDAIR |
931C47FD 00000010 ED0F9A40 03000098 30810000 | 1.....q.a.. |
00000020 E3E2D640 D6E4E3D7 E4E340C3 D740C5E2 | TSO OUTPUT CP ES |
00000030 E3C1C540 40404040 40404040 40404040 | TAE |
00000040 40404040 40404040 40404040 40404040 | |
00000050 40404040 40404040 40404040 40404040 | |
00000060 40404040 40404040 40404040 40404040 | |
00000070 40404040 40404040 40404040 40404040 | |
00000080 40404040 00000000 00000000 00000000 | ..... |
00000090 00000000 00000000 00000000 00000000 | ..... |
000000A0 00000000 00000000 00000005 000BDE10 | ..... |
000000B0 000BDBEC 0000003B 00000000 FFFFB78 | ..... |
000000C0 000BDD8 007C4B70 007C8658 007F1790 | ...Q.@...@f... |
000000D0 00006FE8 000BDD14 400B8F58 000BDE88 | ..?Y...."....h |
000000E0 700B91BA 00000000 5EB0EE40 01D3D07F | ..j.....;".L " |
000000F0 00000000 C000001D 0000001D 81C17000 | .....aA.. |
00000100 FE000000 01D3D07F 00000000 00000000 | .....L "..... |
00000110 00000000 00000000 01C3D96F 00000000 | .....CR?.... |
00000120 DF881C02 00000000 070C3000 000B9204 | .h.....k. |
00000130 01400060 00000000 00000000 01E80348 | .-.....Y.. |
00000140 00FCE210 000010DA 0042001D 00087B69 | ..S.....#. |

```

Figure 29. IPCS Storage Panel Browse Option Pointer Panel

The storage panel provides both a hexadecimal and a character representation of the source data. Each line of the storage display represents either 16 or 32 bytes of storage, depending on the size of the terminal display screen.

- To display 16 bytes of data per line, the minimum line size on the terminal display must be less than 136.
- To display 32 bytes of data per line, the minimum line size on the terminal display must be greater than or equal to 136.
- The storage panel initially shows character data as EBCDIC text. Use the ASCII primary command to change that to show character data as ISO-8 ASCII characters. ASCII persists until the EBCDIC primary command restores the original interpretation.

Scrolling Through the Storage Panel

Several techniques can be used for scrolling:

- Scrolling can be accomplished using the UP and DOWN primary commands that are entered manually or by using PF keys.
- Selection codes **%** and **?** can be used to scroll to storage addressed by a word of formatted storage.
- The LOCATE primary command can be used to scroll to an address.

Selecting Words from the Storage Panel

The following selection codes request IPCS to:

- Interpret the word as an address in the current address space
- Place a pointer for the word in the pointer stack on the pointer panel

For more information about the codes, see [*z/OS MVS IPCS Commands*](#).

L

Interpret the word as a 24-bit address

H

Interpret the word as a 31-bit address

%

Interpret the word as a 24-bit address and display the addressed storage

?

Interpret the word as a 31-bit address and display the addressed storage

!

Interpret the double word as a 64-bit address and display the addressed storage

These selection codes perform what is known as a *fast stack*. They provide the same function as the STACK subcommand in a simpler form.

Figure 30 on page 52 shows how you would enter the %, H, and L selection codes on the storage panel to reference the addresses on the pointer panel.

| ASID(X'0001') STORAGE ----- | | | | | | SCROLL ==> CSR |
|-----------------------------|----------|---|----------|------------|----------|------------------|
| COMMAND ==> | | | | | | |
| 00FCE1F0. | 00FE1818 | % | 00000000 | C4D9C9E5 | C5D940D4 |DRIVER M |
| 00FCE200. | F4404040 | | 40404040 | 00003081 | F0F3F840 | 4a038 |
| 00FCE210. | 00000218 | | 00FF9F80 | L 00FCDFE4 | 00000000 |U.... |
| 00FCE220. | 00000000 | | 00FE88BC | 00FF412E | 00FFF7F6 |h".....76 |
| 00FCE230. | 00FFF76C | | 00000000 | 00000000 | 00FE2B32 | ..7%..... |
| 00FCE240. | 00F9F980 | | 00FFC6A8 | 0082004F | 00FDD240 | .99...Fy.b. ...K |
| 00FCE250. | 00FD1CC8 | | 00FE36E8 | 00000000 | 00000000 | ...H...Y..... |
| 00FCE260. | 0A0307FE | | 00FCDFEC | 00FCE050 | 00000000 |&.... |
| 00FCE270. | 00000000 | | 00FCEB28 | 00FE64EC | 00FE650C |&.... |
| 00FCE280. | 00FCC550 | | 93FD5B28 | 00000000 | 00FFF370 | ..E&1\$.....3. |
| 00FCE290. | 00000000 | H | 81001428 | 00FF7734 | 00000000 |a..... |
| 00FCE2A0. | 00000000 | | 00FDD240 | 00FE5138 | 00000000 |K..... |
| 00FCE2B0. | 00000000 | | 7FFFFFFF | 00000000 | 00000000 |&.... |
| 00FCE2C0. | 00FD8B08 | | 0000AB20 | 00FFB240 | 00FCE020 |&.... |
| 00FCE2D0. | 00FDD570 | | 80F7B2C8 | 00FCE7E0 | E0000000 | ..N..7"H..X".... |
| 00FCE2E0. | 00000000 | | 0A0D0A06 | 00000000 | 00FCE908 |Z..... |
| 00FCE2F0. | 00FCE8A8 | | 00000000 | 00FFF4A0 | 00FE20A0 | ..Yy.....4".... |
| 00FCE300. | 00000000 | | 80000000 | 00000000 | 00FCE750 |X&.... |
| 00FCE310. | 00F89788 | | 00000000 | 80000000 | 010D7310 | .8ph..... |
| 00FCE320. | 00DD7000 | | 00FFC1F0 | 00000000 | 00087000 |A0..... |
| 00FCE330. | 00FE4AA2 | | 00000000 | 00FDD448 | 00000000 | ...s.....M..... |

Figure 30. Entering the %, H, and L Selection Codes on the Browse Option Storage Panel

After ENTER is pressed on the preceding panel, the Browse option scrolls to the requested address of storage (as indicated by the % selection code) and adds stack entries to the pointer stack (as indicated by the %, H, and L selection codes). Figure 31 on page 52 and Figure 32 on page 53 show the results of processing selection codes, %, H, and L.

Figure 31 on page 52 illustrates how the selection code, %, scrolled to the requested address of storage (00000000). Figure 32 on page 53 shows the results of processing the %, H, and L selection codes. Notice the addition of three new pointer entries:

- Pointer 00006 is added to the pointer stack with address 00000000 for address space 1.
- Pointer 00007 is added to the pointer stack with address 00FCDFE4 for address space 1.
- Pointer 00008 is added to the pointer stack with address 01001428 for address space 1.

| ASID(X'0001') STORAGE ----- | | | | | SCROLL ==> CSR |
|-----------------------------|---------------------------------------|----------|----------|----------|------------------|
| COMMAND ==> | | | | | |
| 00000000. | 040C0000 | 810A2150 | 00000000 | 40000000 |a.&.... |
| 00000010. | 00FD2C30 | 00000000 | 070E0000 | 00000000 |&.... |
| 00000020. | 070C1000 | 819DA12C | 040C0000 | 810AFFE0 |a.....a.... |
| 00000030. | 00000000 | 00000000 | 070E0000 | 00000000 |&.... |
| 00000040. | 00000000 | 00000000 | 00000000 | 00FD2C30 |&.... |
| 00000050. | 00000000 | 00000000 | 040C0000 | 810BAF20 |a..... |
| 00000060. | 040C0000 | 80FE3480 | 000C0000 | 81FA1170 |a..... |
| 00000070. | 00080000 | 81FA1A70 | 040C0000 | 80FE3080 |a..... |
| 00000080. | 00000000 | 00001202 | 0002008A | 00040016 |&.... |
| 00000090. | 0022F000 | 00000000 | 00000000 | 00000000 | ..0..... |
| 000000A0. | 00000000 | 00FF4000 | 00000000 | 00000000 |4..... |
| 000000B0. | 00000000 | 00000000 | 0001014C | 00FBB888 |<...h |
| 000000C0. | LENGTH(320)==>All bytes contain X'00' | | | | |
| 00000200. | D7E2C140 | 00020042 | 00F8ADB8 | 01E29DB8 | PSA8[...S.. |
| 00000210. | 00F918A0 | 01C648A0 | 008F6B00 | 008F6B00 | .9...F..... |
| 00000220. | 00F97400 | 00F97400 | 00000000 | 00000000 | .9...9..... |
| 00000230. | LENGTH(32)==>All bytes contain X'00' | | | | |
| 00000250. | 040C0000 | 00A91594 | 040C0000 | 819BE0E0 |z.m....a... |
| 00000260. | AD000950 | AD040950 | AD040950 | AD070950 | [..&[...&[...& |
| 00000270. | AD000950 | 00000000 | 00000404 | 00000000 | [..&[...&[...& |

Figure 31. Result of Processing the % Selection Code on the Browse Option Storage Panel

```

DSNAME('D83DUMP.DUMPC.PB00465') POINTERS -----
COMMAND ==> -                                SCROLL ==> CSR
PTR   Address Address space                   Data type
00001 00000000 HEADER                         AREA
      Remarks: COMMENT 1
00002 00FCE210 ASID(X'0029')                   AREA
      Remarks: COMMENT 2
00003 00000DB0 ASID(X'0008')                   AREA
      Remarks:
00004 00F9AC68 ASID(X'0029')                   AREA
      Remarks:
00005 00FCE1F8 ASID(X'0001')                   STRUCTURE (CVT)
      Remarks: COMMUNICATIONS VECTOR TABLE
00006 00000000 ASID(X'0001')                   AREA
      Remarks:
00007 00FCDFE4 ASID(X'0001')                   AREA
      Remarks:
00008 01001428 ASID(X'0001')                   AREA
      Remarks:
***** BOTTOM OF DATA *****

```

Figure 32. Pointer Panel Result of Processing the %, H, and L Selection Codes on the Browse Option Pointer Panel

Storage Summary Comments

One-line summaries appear on the storage display panel for these situations:

- **Storage not available [to end of qualifiers]**

This comment states that a block of storage cannot be obtained from a dump. If that block extends to the end of the address space being viewed, the phrase “to the end of **qualifiers**” is added. The **qualifiers** are one or two address processing parameters that indicate which address space is being viewed.

- **All bytes contain X'xx'[, C'c']**

This comment states that a single character value is repeated in enough successive bytes to fill one or more standard display lines. If an EBCDIC-graphic is supplied for the value on either the 1403 TN print chain or the 3211 T11 print train, the optional “C'c'” is included.

- **Same as above**

This comment states that the preceding line of data is repeated.

Note: If a summary line does not fill the final line in a display, dump storage (or another summary line) appears on the following line.

Figure 33 on page 54 shows the storage summary comments that might appear in your dump data after selecting the third pointer from the pointer panel (see Figure 32 on page 53). Address space 8 appears in the upper-left corner and the first address referenced is 00000DB0. Starting at address DC0, 576 bytes of dump data contain X'00'. Starting at address 1000, 52992 bytes of dump data are not available for viewing. Starting at address E240, 144 bytes are a repetition of the preceding dump data.

```

ASID(X'0008') STORAGE -----
COMMAND ==>
00000DB0. 01937D80 00FB6EF0 0194E7F8 00937E20 | .1'...>0.mX8.1=. |
00000DC0. LENGTH(576)==>All bytes contain X'00'
00001000. LENGTH(52992)==>Storage not available
0000DF00. D4E3C1E3 C6E2E6C1 D7E2E640 D9C5C7E2 | MTATFSWAPSW REGS |
0000DF10. 00029D60 0000E6B0 000CF0C3 00010010 | ...-..W...0C... |
0000DF20. 2FB00060 000009CF B0000000 0A00008C | ...-.....ó |
0000DF30. 000008D1 E2F300DD A000D7E3 C3C8AF7E | ...JS3...PTCH.= |
0000DF40. AF80AF82 AF84AF86 AF88AF8A AF8CAF8E | ...b.d.f.h.. |
0000DF50. AF90AF92 AF94AF96 AF98AF9A AF9CAF9E | ...k.m.o.q....' |
0000DF60. AFA0AFA2 AFA4AFA6 AFA8AFAA AFACAFAE | ...s.u.w.y.....ó |
0000DF70. AFB0AFB2 AFB4AFB6 AFB8AFBA AFBCAFBE | .....0 |
0000DF80. AFC0AFC2 AFC4AFC6 AFC8AFCA AFCCAFCF | ...B.D.F.H..... |
0000DF90. LENGTH(16)==>All bytes contain X'00'
0000DFA0. LENGTH(16)==>All bytes contain X'40', C' '
0000DFB0. C9C1E3C1 C2D4D540 . FF000000 00000000 | IATABMN ..... |
0000DFC0. LENGTH(624)==>All bytes contain X'00'
0000E230. 00000000 00000000 C6E2D3C7 C1C5D5C4 | .....FSLGAEND |
0000E240. LENGTH(144)==>Same as above
0000E2D0. C9C1E3D5 E4C34040 D3D6E660 C3D6D9C5 | IATNUC LOW-CORE |
0000E2E0. C8C960C3 D6D9C540 C9C1E3D9 D1D4D540 | HI-CORE IATRJMNM |
0000E2F0. 0002D7E3 C3C8F1C6 F1C8F1CA F1CCF1CE | ..PTCH1F1H1.1.1. |

```

Figure 33. Storage Summary Comments on the Browse Option Storage Panel

Using Primary Commands on the Storage Panel

Primary commands can be entered on the command line of the storage panel. [z/OS MVS IPCS Commands](#) contains a list of the valid commands and PF keys for this panel.

There are two special symbols, CURSOR and X, that are accepted by the Browse option on the storage panel when entering the IPCS, LOCATE, and STACK primary commands. These special symbols associate a location in a dump and are used in the same manner as other symbols, such as the CVT and TCB symbols.

- **CURSOR** indicates the word of storage at which you position the cursor. By placing the cursor in the selection field preceding a word of storage or by placing the cursor under a word of storage, you can reference the word of storage. CURSOR is not in effect if the position of the cursor does not identify a word of storage or if you leave the storage panel.
- **X** indicates the starting address of the data displayed on the storage panel.

Note:

1. X remains in effect even if you leave the storage panel.
2. If you issue the IPCS primary command with a subcommand that changes X, the storage displayed will be updated to reflect the location of the changed X. For example, the storage of the IEFBR14 module is displayed if you enter the following on the storage panel:

```
ipcs findmod iefbr14
```

Figure 34 on page 55 shows the use of the special symbol CURSOR, in conjunction with the % selection code, on the STACK and LOCATE primary commands. The figure was shown after selecting the third pointer in the pointer stack on the pointer panel for address space 8 (see Figure 32 on page 53). On the command line, the two primary commands are entered, separated by a semicolon.

The **stack cursor%** primary command requests that an entry to the stack on the pointer panel be added with the address contained in the word of storage indicated by the cursor's current position. The % selection code indicates that this is a low precision (24 bit) address of storage and therefore, the first byte of the address is dropped.

The **locate cursor%** primary command requests that IPCS locate and display the data found at the address contained in the word of storage indicated by the cursor's current position.


```

ASID(X'0008') STORAGE -----
COMMAND ==> stack cursor%;locate cursor%          SCROLL ==> CSR
00000DB0. 01937D80 00FB6EF0 0194E7F8 00937E20 | .1'...>0.mX8.l=. |
00000DC0. LENGTH(576)==>All bytes contain X'00'
00001000. LENGTH(52992)==>Storage not available
0000DF00. D4E3C1E3 C6E2E6C1 D7E2E640 D9C5C7E2 | MTATFSWAPSW REGS |
0000DF10. 00029D60 0000E6B0 000CF0C3 00010010 | ...-..W...0C.... |
0000DF20. 2FB00060 000009CF B0000000 0A00008C | ...-00000000.... |
0000DF30. 000008D1 E2F300DD A000D7E3 C3C8AF7E | ...JS3....PTCH.= |
0000DF40. AF80AF82 AF84AF86 AF88AF8A AF8CAF8E | ...b.d.f.h...d.. |
0000DF50. AF90AF92 AF94AF96 AF98AF9A AF9CAF9E | ...k.m.o.q.....' |
0000DF60. AFA0AFA2 AFA4AFA6 AFA8AFAA AFACAFAE | ...s.u.w.y.....ò |
0000DF70. AFB0AFB2 AFB4AFB6 AFB8AFBA AFBCAFBE | .....0 |
0000DF80. AFC0AFC2 AFC4AFC6 AFC8AFCA AFCCAFCF | ...B.D.F.H..... |
0000DF90. LENGTH(16)==>All bytes contain X'00'
0000DFA0. LENGTH(16)==>All bytes contain X'40', C' '
0000DFB0. C9C1E3C1 C2D4D540 FF000000 00000000 | IATABMN ..... |
0000DFC0. LENGTH(624)==>All bytes contain X'00'
0000E230. 00000000 00000000 C6E2D3C7 C1C5D5C4 | .....FSLGAEND |
0000E240. LENGTH(144)==>Same as above
0000E2D0. C9C1E3D5 E4C34040 D3D6E660 C3D6D9C5 | IATNUC LOW-CORE |
0000E2E0. C8C960C3 D6D9C540 C9C1E3D9 D1D4D540 | HI-CORE IATRJMNM |
0000E2F0. 0002D7E3 C3C8F1C6 F1C8F1CA F1CCF1CE | ..PTCH1F1H1.1.1. |

```

Figure 34. Using Primary Commands on the Browse Option Storage Panel

Figure 35 on page 55 and Figure 36 on page 56 show the results of processing the two primary commands. Figure 35 on page 55 illustrates how the locate command scrolled to the requested address of storage (00000000). Figure 36 on page 56 shows that an additional pointer is added to the pointer stack (00008), with address 00000000 for address space 8.

Note: When data from an address space other than the current one is needed, IPCS allows you to reference it without leaving the storage panel. The address space parameters on the LOCATE and STACK primary commands provide you with this capability.

```

ASID(X'0008') STORAGE -----
COMMAND ==>          SCROLL ==> CSR
00000000. 040C0000 810A2150 00000000 40000000 | ...a.&;... |
00000010. 00FD2C30 00000000 070E0000 00000000 | ..... |
00000020. 070C1000 819DA12C 040C0000 810AFFE0 | ...a.....a... |
00000030. 00000000 00000000 070E0000 00000000 | ..... |
00000040. 00000000 00000000 00000000 00FD2C30 | ..... |
00000050. 00000000 00000000 040C0000 810BAF20 | .....a... |
00000060. 040C0000 80FE3480 000C0000 81FA1170 | .....a... |
00000070. 00080000 81FA1A70 040C0000 80FE3080 | ...a..... |
00000080. 00000000 00001202 0002008A 00040016 | ..... |
00000090. 0022F000 00000000 00000000 00000000 | ..0..... |
000000A0. 00000000 00FFF400 00000000 00000000 | .....4..... |
000000B0. 00000000 00000000 0001014C 00FBB888 | .....<...h |
000000C0. LENGTH(320)==>All bytes contain X'00'
00000200. D7E2C140 00020042 00F8ADB8 01E29DB8 | PSA .....8[.S.. |
00000210. 00F918A0 01C648A0 008F6B00 008F6B00 | .9...F..... |
00000220. 00F97400 00F97400 00000000 00000000 | .9...9..... |
00000230. LENGTH(32)==>All bytes contain X'00'
00000250. 040C0000 00A91594 040C0000 819BE0E0 | ....z.m....a... |
00000260. AD000950 AD040950 AD040950 AD070950 | [..&[..&[..&[..& |
00000270. AD000950 00000000 00000404 00000000 | [..&;..... |

```

Figure 35. Result of Processing the LOCATE Primary Command on the Browse Option Storage Panel

```

DSNAME('D83DUMP.DUMPC.PB00465') POINTERS -----
COMMAND ==> -                                SCROLL ==> CSR
PTR   Address Address space                   Data type
00001 00000000 HEADER                         AREA
      Remarks: COMMENT 1
00002 00FCE210 ASID(X'0029')                   AREA
      Remarks: COMMENT 2
00003 00000DB0 ASID(X'0008')                   AREA
      Remarks:
00004 00F9AC68 ASID(X'0029')                   AREA
      Remarks:
00005 00FCE1F8 ASID(X'0001')                   STRUCTURE (CVT)
      Remarks: COMMUNICATIONS VECTOR TABLE
00006 00FCDFE4 ASID(X'0001')                   AREA
      Remarks:
00007 00FCDFE4 ASID(X'0001')                   AREA
      Remarks:
00008 01001428 ASID(X'0001')                   AREA
      Remarks:

```

Figure 36. Result of Processing the STACK Primary Command on the Browse Option Pointer Panel

Examining 64-bit Dump Data Sets

IPCS supports dump data sets in 64-bit mode. The following examples show output that can be expected when examining dumps that contain 64-bit addresses.

Issue the “ld” option to show the following LISTDUMP output.

```

Source of Dump                                Blocks      Bytes
DSNAME('D83DUMP.SYS0510.SA00459') . . . . . 503,976 . . 2,096,540,160
ABSOLUTE
 00.:01_4FFFFFFF.
X'01_50000000' bytes described in ABSOLUTE

CPU(X'00') STATUS
 00.:0FFF.
X'1000' bytes described in CPU(X'00') STATUS

CPU(X'01') STATUS
 00.:0FFF.
X'1000' bytes described in CPU(X'01') STATUS

CPU(X'02') STATUS
 00.:0FFF.
X'1000' bytes described in CPU(X'02') STATUS

CPU(X'03') STATUS

```

Figure 37. LISTDUMP Output in 64-bit Dump

Issue the “br” option to show the following BROWSE output for the same dump. The result is the Pointer Panel.

```

DSNAME('D83DUMP.SYS0510.SA00459') POINTERS -----
Command ==>                                SCROLL ==> CSR
CPU(X'00') ASID(X'0001') is the default address space
PTR   Address Address space                   Data type
0001 00000000 CPU(X'00') ASID(X'0001')       AREA
      Remarks:
***** END OF POINTER STACK *****

```

Figure 38. Pointer Panel for the Browse Option

Display the data at location 1_00000000 to see data in 64-bit mode.

```

CPU(X'00') ASID(X'0001') ADDRESS(00.) STORAGE -----
Command ==> 1 1_00000000 absolute SCROLL ==> CSR

00000000 000A0000 000130E1 55555555 55555555 ] ..... ]
00000010 00FCEE18 55555555 55555555 55555555 ] ..... ]
00000020.:3F.--All bytes contain X'55'
00000040 55555555 55555555 55555555 00FCEE18 ] ..... ]
00000050 55555555 55555555 000A0000 000140E1 ] ..... ]
00000060 000A0000 000150E1 000A0000 000160E1 ] .....&..... ]
00000070 000A0000 000170E1 000A0000 000180E1 ] ..... ]
00000080 55555555 00031202 0002002F 00040016 ] ..... ]
00000090 00000000 55555555 55555555 55555555 ] ..... ]
000000A0 02000001 0128BC08 00000000 0023B000 ] ..... ]
000000B0 55555555 55555555 000100EE 00E35770 ] .....T.. ]
000000C0 28000000 00000000 E0000000 55555555 ] .....\...... ]
000000D0.:EF.--All bytes contain X'55'
000000F0 55555555 55555555 55555555 00000000 ] ..... ]
00000100.:012F.--All bytes contain X'55'
00000130 07060000 00000000 00000000 00000000 ] ..... ]
00000140 07040000 80000000 00000000 7F599D22 ] ..... ".... ]
00000150 0404E000 80000000 00000000 00FECF1E ] ..\..... ]
00000160.:016F.--All bytes contain X'55'
00000170 07040000 80000000 00000000 01542AE4 ] .....U ]
00000180.:019F.--All bytes contain X'00'
000001A0 04040000 80000000 00000000 0128FD10 ] ..... ]

```

Figure 39. Displaying a 64-bit Location

Line mode information is displayed in a style similar to the BROWSE option. Enter the LIST subcommand.

```

----- IPCS Subcommand Entry -----
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation below:

==> 1 x length(x'92110')

----- IPCS Subcommands and Abbreviations -----
ADDUMP          ] DROPDUMP, DROPD ] LISTMAP, LMAP ] RUNCHAIN, RUNC
ANALYZE         ] DROPMAP, DROPM ] LISTSYM, LSYM ] SCAN
ARCHECK         ] DROPSYM, DROPS ] LISTUCB, LISTU ] SELECT
ASCBEXIT, ASCBX ] EQUATE, EQU, EQ ] LITERAL        ] SETDEF, SETD
ASMCHECK, ASMK  ] FIND, F          ] LPAMAP         ] STACK
CBFORMAT, CBF   ] FINDMOD, FMOD    ] MERGE          ] STATUS, ST
CBSTAT          ] FINDUCB, FINDU   ] NAME           ] SUMMARY, SUMM
CLOSE           ] GTFTRACE, GTF     ] NAMETOKN       ] SYSTRACE
COPYDDIR        ] INTEGER          ] NOTE, N        ] TCBEXIT, TCBX
COPYDUMP        ] IPCS HELP, H     ] OPEN           ] VERBEXIT, VERBX
COPYTRC         ] LIST, L          ] PROFILE, PROF  ] WHERE, W
CTRACE          ] LISTDUMP, LDMP    ] RENUM, REN     ]

```

Figure 40. Using the LIST Subcommand for a 64-bit Dump

The following output is displayed.

```

IPCS OUTPUT STREAM ----- Line 0 Cols 1 78
Command ==>                SCROLL ==> CSR
***** TOP OF DATA *****

LIST 00000001_00000000. ABSOLUTE LENGTH(X'092110') AREA
01_00000000. LENGTH(X'092000')==>All bytes contain X'00'
_0092000. 00000001_00091058 00000001_00090058 ].....]
_0092010. 00000001_0008F058 00000001_0008E058 ].....0.....\.]
_0092020. 00000001_0008D058 00000001_0008C058 ].....}.....{.]
_0092030. 00000001_0008B058 00000001_0008A058 ].....]
_0092040. 00000001_00089058 00000001_00088058 ].....]
_0092050. 00000001_00087058 00000001_00086058 ].....-.]
_0092060. 00000001_00085058 00000001_00084058 ].....&.....]
_0092070. 00000001_00083058 00000001_00082058 ].....]
_0092080. 00000001_00081058 00000001_00080058 ].....]
_0092090. 00000001_0007F058 00000001_0007E058 ].....0.....\.]
_00920A0. 00000001_0007D058 00000001_0007C058 ].....}.....{.]
_00920B0. 00000001_0007B058 00000001_0007A058 ].....]
_00920C0. 00000001_00079058 00000001_00078058 ].....]
_00920D0. 00000001_00077058 00000001_00076058 ].....-.]
_00920E0. 00000001_00075058 00000001_00074058 ].....&.....]
_00920F0. 00000001_00073058 00000001_00072058 ].....]
_0092100. 00000001_00071058 00000001_00070058 ].....]
***** END OF DATA *****

```

Figure 41. Output from the LIST Subcommand - 64-bit Dump - Example 1

IPCS supports UNSIGNED and POINTER data from 1-8 bytes in length and supports SIGNED data in 2-byte, 4-byte, and 8-byte lengths. Pointer data longer than 4 bytes in length is shown with an underscore linking the first hexadecimal digits and the final 8 hexadecimal digits. A two-column gutter between pointers helps distinguish one pointer from the next shown on a line.

The following output results from issuing the subcommand

```

1 1_00092000 absolute pointer length(8) dimension(34)

```

```

IPCS OUTPUT STREAM ----- Line 0 Cols 1 78
Command ==>                SCROLL ==> CSR
***** TOP OF DATA *****

LIST 00000001_00092000. ABSOLUTE LENGTH(X'08') ENTRIES(X'+01':X'+22') POINTER
_0092000 00000001_00091058 00000001_00090058 00000001_0008F058
_0092018 00000001_0008E058 00000001_0008D058 00000001_0008C058
_0092030 00000001_0008B058 00000001_0008A058 00000001_00089058
_0092048 00000001_00088058 00000001_00087058 00000001_00086058
_0092060 00000001_00085058 00000001_00084058 00000001_00083058
_0092078 00000001_00082058 00000001_00081058 00000001_00080058
_0092090 00000001_0007F058 00000001_0007E058 00000001_0007D058
_00920A8 00000001_0007C058 00000001_0007B058 00000001_0007A058
_00920C0 00000001_00079058 00000001_00078058 00000001_00077058
_00920D8 00000001_00076058 00000001_00075058 00000001_00074058
_00920F0 00000001_00073058 00000001_00072058 00000001_00071058
_0092108 00000001_00070058
***** END OF DATA *****

```

Figure 42. Output from the LIST Subcommand - 64-bit Dump - Example 2

IPCS introduces the OPCODE command to help the IPCS user deal with operation codes that may not be familiar when shown using hexadecimal digits.

```

----- IPCS Subcommand Entry -----
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation below:

====> opcode 47f0f01c

----- IPCS Subcommands and Abbreviations -----
ADDUMP      ] DROPDUMP, DROPD ] LISTMAP,  LMAP   ] RUNCHAIN, RUNC
ANALYZE      ] DROPMAP,  DROPM ] LISTSYM, LSYM   ] SCAN
ARCHECK      ] DROPSYM, DROPS ] LISTUCB, LISTU  ] SELECT
ASCBEXIT, ASCBX ] EQUATE, EQU, EQ ] LITERAL          ] SETDEF,  SETD
ASMCHECK, ASMK ] FIND,    F        ] LPAMAP           ] STACK
CBFORMAT, CBF ] FINDMOD, FMODE   ] MERGE            ] STATUS,  ST
CBSTAT       ] FINDUCB, FINDU ] NAME             ] SUMMARY, SUMM
CLOSE        ] GTFTRACE, GTF  ] NAMETOKN         ] SYSTRACE
COPYDDIR     ] INTEGER          ] NOTE,            ] TCBEXIT, TCBX
COPYDUMP     ] IPCS HELP,  H   ] OPEN             ] VERBEXIT, VERBX
COPYTRC      ] LIST,      L    ] PROFILE, PROF    ] WHERE,    W
CTRACE       ] LISTDUMP, LDMP ] RENUM,          ] REN

```

Figure 43. Using the OPCODE Subcommand

The OPCODE command produces the following output.

```

IPCS OUTPUT STREAM ----- Line 0 Cols 1 78
Command ==>                SCROLL ==> CSR
***** TOP OF DATA *****

Mnemonic for X'47F0F01C' is BC
***** END OF DATA *****

```

Figure 44. Output from the OPCODE Subcommand

Getting Online Help in the IPCS Dialog

The IPCS dialog supports an extensive online help facility. Tutorial and help panels provide information on the following topics:

- Using IPCS panels and primary and line commands
- Entering IPCS subcommands
- Reading formatted dump reports on a dump display reporter panel
- Choosing an option from the Dump Component Data Analysis menu

Use the online help facilities to get quick answers to problems in these areas.

Help Using IPCS Panels and Primary and Line Commands

Through the ISPF help facility, IPCS provides a tutorial on using the IPCS dialog. The tutorial answers questions about IPCS primary and line commands and panels.

To access the tutorial, press PF1 or type HELP on the command line. Usually the tutorial will begin with the particular topic you need.

To exit the tutorial, press PF3 or enter END on the command line.

Help Entering IPCS Subcommands

Through its own help facility, IPCS provides online summaries of the function, syntax, and parameters of all IPCS subcommands. This information is similar to that found in the [z/OS MVS IPCS Commands](#).

To access these online summaries, use the IPCS HELP subcommand.

Note: In the IPCS dialog, you must use H, the abbreviated form of the HELP subcommand. Otherwise, ISPF captures the help request and starts the tutorial on the IPCS dialog.

Help Reading Formatted Dump Reports

While reading a formatted dump report on the dump display reporter panel, press PF1 or enter HELP on the command line to see the tutorial on viewing IPCS reports. On this tutorial panel, options 1 through 5 explain how to use the dump display reporter panel. Option 6, if available, gives information on reading the formatted dump report produced by the subcommand you entered.

Help Choosing a Component Analysis Option

The dump component data analysis menu explains how to get summary information for each component option. See [Figure 10 on page 28](#).

ISPF Operations on the IPCS Dialog

When in the IPCS dialog there are several common ISPF operations that you can use:

- Splitting the display screen
- Entering primary and line commands
- Using PF keys

Splitting the Display Screen

Split-screen support is available to permit sharing of the screen by related data. The data can be areas of the same dump or related areas in different dumps.

For example, the pointer stack on the pointer panel for a dump can be updated from multiple ISPF screens. Pointer stack update requests are processed in the order of screen requests. Updates to the pointer panel on one screen are not reflected on other screens until the next transaction is processed on the other screens.

Entering Primary and Line Commands

IPCS uses primary and line commands in the same manner as ISPF uses them. These commands provide browsing and editing capabilities when viewing dump data in full-screen mode.

Enter primary commands on the COMMAND/OPTION line of a panel. Enter line commands by typing the command at the beginning of the output line.

IPCS has its own primary commands. In addition, you can also use ISPF commands, such as CURSOR, HELP, SPLIT, and SWAP.

z/OS MVS IPCS Commands describes the IPCS primary and line commands and lists the ISPF commands you can use on IPCS dialog panels. The *ISPF User's Guide* describes the ISPF commands.

Using Program Function (PF) Keys

Program function (PF) keys invoke specific IPCS and ISPF primary commands. To display the current PF key definitions, type KEYS on the command line and press Enter. To change the PF key assignments, see [“Customizing the IPCS Dialog” on page 60](#).

Customizing the IPCS Dialog

This topic explains how you can change primary command definitions and PF key assignments for the IPCS dialog. For information on customizing other parts of the IPCS dialog, see [z/OS MVS IPCS Customization](#).

Customizing the Primary Command Definitions

There are two sets of primary commands supplied with the IPCS dialog. One set is used solely on the panels of the Browse option; the other set is used on all other panels. You can change these IPCS dialog primary command definitions to your own specifications by following these steps:

1. Ensure that the IPCS data set 'SYS1.SBLSTBL0' is in the data set concatenation with the ISPF ISPTLIB data set.
2. Ensure that your ISPF profile data set is allocated as ISPF ISPTABL data set.
3. Select option 3.9 from the ISPF/PDF Primary Option Menu to access the ISPF command table utility.
4. Determine which set of primary commands you want to change, and use the command table utility screen, as follows:
 - To change the primary command definitions for the pointer and storage panels of the Browse option, after **APPLICATION ID** ===> enter the following:

```
b1s1
```

- To change the primary command definitions accepted on all other panels, after **APPLICATION ID** ===> enter the following:

```
b1sg
```

5. Press Enter.
6. On the command table utility screen, insert definitions of primary commands. For assistance, use the HELP command.
7. Save the changes by entering on the command line:

```
end
```

After performing these steps to change the primary command definitions, ISPF updates the command table and stores this profile in your ISPF profile data set to be retained between ISPF sessions. Therefore, if you changed the primary commands for the BROWSE option, data set member BLSLCMDS is added to your ISPF profile data set. If you changed the primary commands for all IPCS options except BROWSE and TUTORIAL, data set member BLSGCMDS is added to your ISPF profile data set. Deletion of either one of these data set members will reestablish the original IPCS primary command definitions.

Customizing Your Program Function (PF) Keys

There are two sets of PF keys supplied with the IPCS dialog. One set is used solely on the Browse pointer and storage panels; the other set is used on all other panels. To change these PF key definitions to your own specifications follow these steps:

1. Determine which set of PF keys you want to change.
 - To change the PF keys for the pointer and storage panels of the IPCS Browse option, select option 1 (BROWSE) from the IPCS Primary Option Menu panel and enter the following on the COMMAND line:

```
keys
```

This displays the current PF key definitions for the Browse option. If your terminal has 24 PF keys, press Enter again to see the definitions for the remaining keys.

```
PF1  ==>  HELP
PF2  ==>  SPLIT
PF3  ==>  END
PF4  ==>  RETURN
PF5  ==>  RFIND
PF6  ==>  STACK
PF7  ==>  UP
PF8  ==>  DOWN
PF9  ==>  SWAP
PF10 ==>  STACK X;LOCATE %
PF13 ==>  HELP
PF14 ==>  SPLIT
PF15 ==>  END
PF16 ==>  RETURN
PF17 ==>  RFIND
PF18 ==>  STACK
PF19 ==>  UP
PF20 ==>  DOWN
PF21 ==>  SWAP
PF22 ==>  STACK X; LOCATE %
```

```
PF11 ===> STACK X;LOCATE ?    PF23 ===> STACK X; LOCATE ?
PF12 ===> CURSOR              PF24 ===> CURSOR
```

- To change the PF keys accepted on all other panels, enter the following on the OPTION line of the IPCS Primary Option Menu panel:

```
keys
```

This displays the current PF key definitions. If your terminal has 24 PF keys, press Enter again to see the definitions for the remaining keys.

```
PF1  ===> HELP          PF13 ===> HELP
PF2  ===> SPLIT          PF14 ===> SPLIT
PF3  ===> END            PF15 ===> END
PF4  ===> RETURN         PF16 ===> RETURN
PF5  ===> RFIND          PF17 ===> RFIND
PF6  ===> MORE           PF18 ===> MORE
PF7  ===> UP             PF19 ===> UP
PF8  ===> DOWN           PF20 ===> DOWN
PF9  ===> SWAP           PF21 ===> SWAP
PF10 ===> LEFT           PF22 ===> LEFT
PF11 ===> RIGHT          PF23 ===> RIGHT
PF12 ===> CURSOR         PF24 ===> CURSOR
```

2. Read the instructions on the panels to review the current definitions of your PF keys and to change them. For assistance, use the HELP primary command.
3. Save the changes by entering the following on the COMMAND or OPTION line:

```
end
```

After performing these steps to change your PF key definitions, ISPF updates the command table and stores this profile in your ISPF profile data set to be retained between ISPF sessions. Therefore, if you were changing your Browse option PF keys, ISPF adds data set member BLSLPROF to your ISPF profile data set. Likewise, if you were changing your other set of PF keys, ISPF adds data set member BLSGPROF to your ISPF profile data set. Deletion of either one of these data set members will reestablish the original IPCS PF key definitions. However, keep in mind that these data set members also contain information that the IPCS dialog retains between sessions, such as the JOB statement for an IPCS batch job, and deletion of any one of these members will cause information to be lost.

Using Data Privacy for Diagnostics

Data Privacy for Diagnostics Analyzer provides the capability to post process the following memory dump types that are taken on a z15 or later processor:

- SVC
- Stand-alone
- SLIP
- SYSMDUMP (from V2.5)
- Transaction (from V2.5)

Post processing is used to redact pages that have been tagged as being sensitive by the applications that created the pages, as well as untagged pages that will be scanned and detected as containing sensitive data per the Data Privacy for Diagnostics Analyzer, which requires a minimum of IBM 64-bit SDK for z/OS Java™ Technology Edition version 8.0. This redacted version of the original memory dump is written to a new memory dump data set without modifying the original memory dump data set. Retain both memory dumps for as long as it takes to diagnose the reported problem.

Append Dump Directory records (BLSADDIR) are removed when generating a redacted stand-alone memory dump. Additional processing is required for stand-alone memory dumps that contain captured memory dumps. If the captured memory dumps are required by vendors, the memory dumps must first be extracted (IPCS COPYCAPD) from the original stand-alone memory dump, then processed separately. Do not depend on captured memory dumps being available within a redacted stand-alone memory dump.

Note: A stand-alone memory dump can contain one or more SVC memory dumps that are captured in memory, but were not written to a data set. It is recommended that you extract these SVC dumps by using IPCS COPYCAPD, if captured on z15 or later processors, and post-process them before sending them to IBM for further analysis to ensure that sensitive data is properly protected.

The following functions are provided:

REDACT

You can redact any data that is tagged as sensitive=yes without further analysis.

Note: You cannot perform the ANALYZE function on a memory dump that has already been redacted by this process.

You can request this processing by using either:

- IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=Y
- Use sample job SYS1.SAMPLIB(BLSJDPFD)

ANALYZE

Any pages tagged sensitive by the applications that own that data as well as any untagged pages detected as containing sensitive data.

Note: You cannot perform the ANALYZE function on a memory dump that has already been redacted by this process.

You can request this processing by using either:

- IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=N
- Use sample job SYS1.SAMPLIB(BLSJDPA).

REPORT

You can create human readable reports for a memory dump that has been processed by the Data Privacy for Diagnostics Analyzer. These reports, once created, are in the *directory/reports/dump-name/run-number* directory in the file system that is used for DPA processing. You can request this processing by using either:

- IPCS option 5.6, specifying the REPORT function.
- Use sample job SYS1.SAMPLIB(BLSJDPR).

FEEDBACK

You may provide feedback for a memory dump that has been processed by the Data Privacy for Diagnostics Analyzer. After looking through the reports and understanding the pages that have or have not been flagged as sensitive, you can provide feedback to help the Data Privacy for Diagnostics Analyzer improve its sensitive data detection. More information is covered on providing feedback later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing by using either:

- IPCS option 5.6, specifying the FEEDBACK function
- Use sample job SYS1.SAMPLIB(BLSJDPI).

INGEST

You may ingest data to help the Data Privacy for Diagnostics Analyzer determine what sensitive data exists in your environment. Data can be ingested from dictionaries, databases, or other sources. This data is added to the knowledge base information and will be used in future analysis runs. More information is covered on providing ingested data later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing by using either:

- IPCS option 5.6, specifying the INGEST function
- Use sample job SYS1.SAMPLIB(BLSJDPI).

EXTRACT

You can extract any built-in or custom identifiers from the Analyzer to a file so that the user may see the exact criteria for determining the sensitivity of the data with the ANALYZE function. The output file contains either the pattern or entire dictionary depending on the type of identifier to help ensure

that the Data Privacy for Diagnostics Analyzer is correctly marking data as sensitive or nonsensitive. More information is covered on extracting identifiers later in this chapter. After updating configuration files and indicating which identifiers can be written to a file, you can request this processing by using either:

- IPCS option 5.6, specifying the EXTRACT function
- Use sample job SYS1.SAMPLIB(BLSJDPX).

Generally, you want to start by performing the ANALYZE function on a memory dump. This function works only on memory dumps captured on a z15 or later processor. After creating the redacted version of the memory dump, you will want to check the memory dump to understand what has been redacted. Reports are available to help you understand why pages have been redacted. You can look at these reports to see whether the data has been properly identified as sensitive. Some reports are written in concise form and must be formatted by using the REPORT function. After running the REPORT function, you may want to give feedback to Data Privacy for Diagnostics Analyzer regarding some of the data that it either found as sensitive but was not sensitive, or feedback on data that was sensitive but not detected as sensitive. The FEEDBACK function allows you to perform this task. The cycle of ANALYZE / REPORT / FEEDBACK provides a way to train the Data Privacy for Diagnostics processing in order to produce memory dumps with the right level of redaction for your environment.

Another function that can be used is the INGEST function. This allows you to import data from databases and files, and lets you create custom information that can be used by the Data Privacy for Diagnostics Analyzer processing to help identify sensitive data. The creation of custom identifiers that are tailored to an installation's data privacy requirements is imperative to attain the most accurate redaction of SPI (or other sensitive information); far surpassing the redaction with using the generic built-in identifiers provided with the Analyzer.

In order to display the exact criteria that the ANALYZE function is using to determine data sensitivity, one might use the EXTRACT function to write out any built-in or custom identifiers to a file such that when that particular identifier is requested in the ANALYZE configuration, the user knows exactly which tokens or what pattern will be used to mark data as sensitive or nonsensitive.

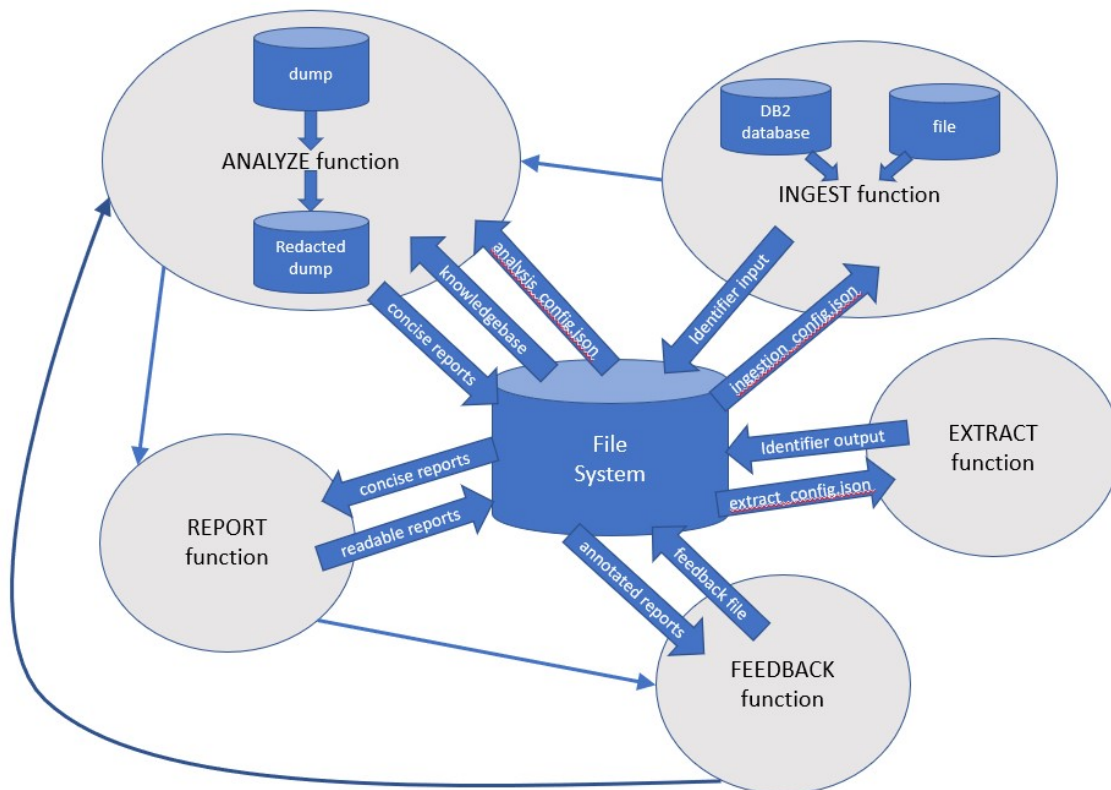


Figure 45. Data Privacy for Diagnostics Usage Cycle

Using the Data Privacy for Diagnostics Analyzer Dialog within IPCS

When IPCS is used, panels are presented to allow you to specify parameters required for processing. The dialog generates appropriate JCL based on the parameters provided. If any data sets are required but not preallocated, the dialog attempts to dynamically allocate them. If dynamic allocation fails for any reason, you should be able to preallocate data sets by using other mechanisms (such as ISPF option 3.2).

Note: Not all parameters are present on all IPCS panels for each function.

The parameters that are specified on the IPCS Data Privacy for Diagnostics Analyzer panels are:

DATA SET NAME

The input memory dump data set name. This option is equivalent to the `input_dataset` parameter in the JCL submitted to perform the requested function.

NEW DATA SET NAME

The output (redacted) memory dump data set name. This option is equivalent to the `output-dump-dataset` field in the JCL submitted to perform the ANALYZE function.

TEMP DATA SET/PAT

Temporary data set names can either be a specific name or a data set name pattern. For more information on patterns, see the help pages. This option is equivalent to the `output_dataset` or `output_dataset_prefix` parameters in the JCL submitted to perform the requested function.

BYPASS DP ANALYSIS

Allows you to submit a job that will either perform analysis (N) or skip analysis (Y). If N is specified, the Data Privacy for Diagnostics Analyzer step scans the input data set looking for additional sensitive data in addition to data identified by the applications that allocated the storage marked as sensitive. If found, either token-level or page level redaction is performed based on the Allow Page Level specification. If Y is specified, this step is bypassed. The output data set identified by the NEW DATA SET NAME field will only remove data that is identified by the applications that allocated the storage marked as sensitive.

REDACTION STRING

If you are not allowing page level redaction, this redaction string is used to overlay data that is determined to be sensitive in the output memory dump. You may leave this field blank to overlay the token with X or specify a string. When longer strings are detected in the pages, the string is used in a repeated fashion. If shorter strings are found, only a portion of the redaction string may be used. This option is equivalent to the `redaction_string` parameter in the JCL submitted to perform the requested function.

NUMBER OF THREADS

For ANALYZE requests, large memory dumps may be processed faster by using multi-threading. You may specify 1 to 8 for the number of threads. Each thread requested processes a portion of the input memory dump, reducing the elapsed time that it takes to process the entire memory dump, however, it may also increase the simultaneous amount of resources that are required to process the request. This option is equivalent to the `thread_count` parameter in the JCL submitted to perform the requested function.

ALLOW PAGE LEVEL

If Y is specified, known as fast-analysis mode or page-level redaction, the entire page of storage is redacted when any sensitive data is detected. Page-level redaction may allow the analysis processing to run faster since processing stops at the first sensitive string in a page is found, however, it is possible that allowing page-level redaction may cause diagnostic data to be lost. If you find this to be true, set the value to N, known as detailed analysis mode or token-level redaction, so that data that is determined to be sensitive will be overlaid by using only the redaction string. The default value is N or token-level redaction.

SENSITIVE REPORT

If Y is specified, reports are generated in `directory/reports/dump-name/runnumber/sensitive_token_log_n` where *n* is the thread number. There will be a file per thread requested. For each string detected, data is written to these files to help you understand what has been redacted and why. Based on this information, you may decide to include or exclude types of data. When

the REPORT function is requested, it consolidates these sensitive_token_log_n files into a human-readable file named sensitive_tokens.

DPfD HOME DIR

Specify the path where the Data Privacy for Diagnostics Analyzer home directory is configured, *directory* as previously described. Do not include the trailing '/' when specifying this path.

JAVA HOME DIR

Specify the path where Java is installed. This is used in the batch job's STDENV set up file to create the proper environment for the Java processing to run in. Do not include the trailing '/' when specifying this path. Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0.

JAVA OPTIONS

You may provide whatever Java options are wanted. For example, you may need to specify a minimum and maximum heap size for the JVM to successfully run a multi-threaded DPfD ANALYZE request. Using the default setup with only built-in identifiers, each thread requires approximately 512 MB to successfully load data for the run. Requesting additional threads or including additional identifiers increase the size of the heap for the JVM, so use the -Xms and -Xmx options to adjust the minimum and maximum heap size. For more information about JVM Command-Line Options, see the topic OpenJ9 command-line options in [IBM SDK, Java Technology Edition 8.0.0](#)

Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0.

If you are using IBM Semeru Runtime Certified Edition for z/OS 21 or later, the file encoding for Data Privacy for Diagnostics files must be specified explicitly via the following parameter due to changes made to the default file encoding: -Dfile.encoding=IBM-1047. For more information, see [IBM Semeru Runtime Certified Edition for z/OS 21 \(www.ibm.com/docs/en/semeru-runtime-ce-z/21?%20%7C%20topic=guide-file-encoding-utf-8-as-default-charset\)](#).

JZOS LOAD MODULE

The dialog uses the JZOS Batch Launcher in the JCL that is submitted. Determine the correct level of JZOS installed on your system and provide the name of the appropriate load module in this parameter. Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0, thus the 64-bit version 8 load module for JZOS Batch Launcher is JVMLDM86. For more information, see the [JZOS Batch Launcher and Toolkit Installation and Users Guide](#).

MIGLIB DATASET

A sort E35 exit is used to remove pages that are flagged as sensitive. This function is provided in module BLSRTE35, which is included in SYS1.MIGLIB. Should you need to override where this exit can be loaded from, provide the name of the MIGLIB that contains the load module you want to run.

TEMP ALLOC PARMS

If your environment requires specific allocation parameters for memory dump data sets, you may supply any allocation parameters that ensure that the data set is properly allocated. For example, supplying DATACLAS and STORCLAS keywords may be necessary to locate the correct storage pool and attributes.

Do not specify RECFM, DSORG, LRECL, BLKSIZE, SPACE, and TRACK as they are used to create some of the interim data sets. If you need to use one of those allocation parameters, request the ANALYZE function by the JCL instead of through IPCS.

EDIT CONFIG FILE?

If Y, allows the user to edit the configuration file pertaining to the function requested (analysis_config.json for ANALYZE or ingestion_config.json for INGEST or extract_config.json for EXTRACT) before submitting the JCL to perform the requested function. Default is N. For more information, see the analysis_config.json, extra_config.json, and ingestion_config.json sections.

RUN NUMBER

From the ANALYZE step, a run number was generated and can be found in the job output, which can be specified for this parameter when the function requested is REPORT or FEEDBACK. If a run number is not specified, the most recent ANALYZE run for the input memory dump is used.

DB2® JDBC PATH

For the INGEST function, if using a Db2 connection source in the `ingestion_config.json` file, this field is needed to specify the path for the Db2 JDBC Driver and License JAR files. Do not include the trailing '/' when specifying this path.

Requesting an ANALYZE run

ANALYZE is the function that looks for sensitive data in memory dump records and flag those records for redaction. If token-level redaction is requested, it overlays that sensitive data with a redaction string. This is accomplished by matching tokens against identifiers and redacting any matches that the Analyzer may detect. A token is a printable character string delimited by a space or non-printable hex data. Token matches are detected as whole strings only, and not as substrings within the dictionary or pattern string of the identifier.

Regardless of how the job is initiated (by IPCS option 5.6 or by the BLSJDPA JCL in SYS1.SAMPLIB), two important configuration files are used:

- Runtime configuration file
 - This file is built by either the dialog by using the supplied parameters, or is specified by an in-stream DD statement in the BLSJDPA JCL.
- `analysis_config.json` file
 - This file provides additional detail on what to include or exclude when looking for sensitive data in memory dump records. It is located in the `/configuration/` directory in the file system. You can use either the EDIT CONFIG FILE option Y in the **ANALYZE IPCS** panel to edit this file if you want to change it, or you can directly edit it using an editor that you are familiar with.

The Runtime configuration is a JSON file that is built by the dialog by using the parameters that are supplied on the panel. It can also be supplied as a file or an instream data set in JCL if you use JCL to submit the job. The runtime configuration file parameters (hand-coded in the BLSJDPA JCL) are:

"input_dataset"

Specifies the location of the input memory dump. Specify a data set name as follows: `"//dump-dataset-name"`.

If using the **ANALYZE IPCS** panel, the **DATA SET NAME** field populates this parameter.

"thread_count"

Specifies the number of worker threads to be created. The total number of threads is one more than this (there is one monitor thread). If omitted, the default value is 4 for the JCL interface. Valid values are 1-8. If using the **ANALYZE IPCS** panel, the **NUMBER OF THREADS** field populates this parameter.

"record_count"

Estimated number of records in the input memory dump. If set to 0 or omitted, the Data Privacy for Diagnostics Analyzer counts the actual number of records. This is used when multiple threads are requested to ensure that the requested number of threads evenly split then records. This parameter is not available in IPCS page interface.

"output_dataset_prefix" or "output_dataset"

Specifies either the prefix to use for each thread's output memory dump data, or the list of data sets. Either data set names or DD names can be specified on this parameter. For example, `"output_dataset_prefix":"SYS1.DUMP.D190926.T132348"` indicates the prefix that is used by each thread. Files are either dynamically allocated or can be pre-allocated as `SYS1.DUMP.D190926.T132348.F1`, `SYS1.DUMP.D190926.T132348.F2`, `SYS1.DUMP.D190926.T132348.F3`, and so on, one per thread. When using the BLSJDPA JCL, you can also specify DDNAMEs as prefixes. For example, you can specify `"output_dataset_prefix":"//DD:ANLZO"` and specify DD statements for `//ANLZOF1`, `//ANLZOF2`,

and so on, for each thread's output. Alternatively, you can use the **"output_dataset"** method of supplying a list of data sets. For example, you can specify "output_dataset": ["//SYS1.DUMP.D190926.T132348.F1","//SYS1.DUMP.D190926.T132348.F2"]. If you are using the dialog to initiate the job, you can also specify a pattern to be used. In this case, the pattern can contain a single "%" character, which causes the dialog to generate data set names with thread numbers substituted in that position in the data set name. For example, you can specify 'SYS1.DUMP.D190926.T132348.F%' as the data set name pattern on the panel and the dialog would generate "output_dataset": ["//SYS1.DUMP.D190926.T132348.F1","//SYS1.DUMP.D190926.T132348.F2"] as the parameters.

If using the **ANALYZE IPCS** panel, the **TEMP DATA SET/PAT** field populates this parameter.

"redaction_string"

String to use for redaction for pages being analyzed by using detailed analysis. When a **"redaction_string"** isn't specified and detailed analysis is specified, sensitive data is replaced with X. When longer strings are detected in the pages, the string is used in a repeated fashion. If shorter strings are found, only a portion of the redaction string can be used. If using the **ANALYZE IPCS** panel, this parameter is populated by the **REDACTION STRING** field.

"analysis_mode"

Specifies the analysis mode for detecting sensitive data. Valid values are 1 for Page-Level Redaction and 2 for Token-Level Redaction. In Page-Level Redaction, the entire page is marked as sensitive when first sensitive token is identified in the page. In Token-Level Redaction, each sensitive token is identified independently and overlaid with the **"redaction_string"**. When 1 is specified, some pages can be analyzed by using Token-Level Redaction. If using the **ANALYZE IPCS** panel, the **ALLOW PAGE LEVEL** field populates this parameter, with Y being equivalent to 1 (Page-Level Redaction) and N being equivalent to 2 (Token-Level Redaction).

"log_sensitive_tokens"

Specifies whether the sensitive token log for each file should be generated. When set to TRUE, a sensitive token log for each thread should be generated in the /reports folder. Valid values are TRUE and FALSE. If using the **ANALYZE IPCS** panel, the **SENSITIVE REPORT** field populates this parameter, with Y being equivalent to TRUE and N being equivalent to FALSE. NOTE: by specifying TRUE, an additional file is generated on each ANALYZE function request, thus the Data Privacy for Diagnostics Analyzer home directory fills up more quickly.

"dpfd_home"

Specifies the Data Privacy for Diagnostics Analyzer home directory. If using the **ANALYZE IPCS** panel, the **DPFD HOME DIR** field populates this parameter.

"character_set"

Specified the character set that should be used for decoding of the input memory dump. The default value is "Cp1047". This parameter is not available in the IPCS panel interface.

Valid values are Cp1047 and US-ASCII.

The analysis_config.json file:

As your experience with this ANALYZE processing matures, this file likely becomes stable until major changes in data occur in your environment. You likely start with the default file that is supplied with the product. As your usage evolves, you can decide to exclude or include certain built-in identifiers, use custom identifiers, or add dependent identifiers. This file is in a JSON format keyword-value pairs and arrays that are used to specify parameters to the ANALYZE function.

Parameter Descriptions

"built_in_identifiers_include"

Specifies the built-in identifiers, which should be used to detect sensitive tokens. Values that can be supplied here are listed in the table below. If nothing is specified, no identifiers are included. These built-in identifiers are specified in a comma-separated list with quotations around the identifier:

```
"built_in_identifiers_include" : [
  "Identifier1",
  "Identifier2",
```

```
...
"IdentifierN"
],
```

see the [Figure 46 on page 71](#).

"built_in_identifiers_exclude"

Specifies the built-in identifiers that should not be used to detect type of tokens. Values that can be supplied here are listed in the table below. If nothing is specified, no identifiers are excluded. These built-in identifiers are specified in a comma-separated list with quotations around the identifier as shown in [Figure 46 on page 71](#).

"custom_identifiers"

Specifies any custom identifiers, which should be used to detect sensitive data. Custom identifiers can be ingested by using the INGEST function and can be in the form of dictionaries or patterns. Arrays of multiple identifiers can be specified with the attributes of a single custom identifier that is enclosed in { } and the identifiers themselves are separated by commas as shown in [Figure 46 on page 71](#). For more information about creating custom identifiers, see ["Requesting an INGEST run" on page 75](#). Each identifier has the following fields:

"format"

The valid value is "custom". "custom" indicates that a previously ingested dictionary or set of patterns is specified.

"inputfilename"

File name containing the identifier data. If the format is "custom", the Data Privacy for Diagnostics Analyzer looks for the file in /knowledgebase/ingested/.

"entitytype"

Specifies a name for the identifier. This is displayed in files that are generated by a REPORT request. If the **"inputsource"** of the identifier is **"csv"** or **"query"** in the `ingestion_config.json` file, the column name is to be used as **"entitytype"** and this specification is ignored.

For all other identifiers, if the entity name is provided when the file is ingested (for "custom" format), this field can be skipped. If values are provided both during ingestion and in `analysis_config.json`, the value that is provided in `analysis_config.json` takes precedence. If no value is provided either during ingestion or in `analysis_config.json`, a custom value is chosen.

"description"

Specifies a description of the identifier. If values are provided both during ingestion and in `analysis_config.json`, the value that is provided in `analysis_config.json` takes precedence. If no value is provided either during ingestion or in `analysis_config.json`, a custom value is chosen.

"dependent_identifiers"

Specifies a set of identifiers, which are sensitive only when they all occur in a page. Here, you specify an array where multiple dependent identifiers can be specified with the attributes of a single dependency that is enclosed in { } and the dependencies themselves are separated by commas as shown in [Figure 46 on page 71](#). When an identifier is made part of **"dependent_identifiers"**, it stops being an independent identifier. All of the identifiers that are specified in a **"dependent_identifiers"** should be present in **"built_in_identifiers_include"** or in **"custom_identifiers"**; otherwise, this is never detected. Each dependent identifier has the following fields:

"name"

Specifies the name that is assigned to this dependent identifier.

"identifiers"

Specifies the set of identifiers that belong to this dependent identifier. These identifiers are specified in a comma-separated list with quotations around the identifier as shown in [Figure 46 on page 71](#).

“built_in_ns_identifiers_include”

Specifies the built-in identifiers, which should be used to detect that a token is nonsensitive. These built-in identifiers are specified in a comma-separated list with quotations around the identifier as shown in Figure 46 on page 71.

“built_in_ns_identifiers_exclude”

Specifies the built-in identifiers, which should not be used to detect that a token is nonsensitive. These built-in identifiers are specified in a comma-separated list with quotations around the identifier as shown in Figure 46 on page 71.

“custom_ns_identifiers”

Specifies any custom identifiers, which should be used to identify tokens as insensitive data. Custom identifiers can be ingested by using the INGEST function and can be in the form of dictionaries, patterns with the attributes of a single custom identifier that is enclosed in { } and the identifiers themselves are separated by commas as shown in Figure 46 on page 71. Arrays of multiple identifiers can be specified. For more information about creating custom identifiers, see “Requesting an INGEST run” on page 75. Each identifier has the following fields:

“format”

The valid value is “custom”. “custom” indicates that a previously ingested dictionary or set of patterns is specified.

“inputfilename”

File name that contains the identifier data. The Data Privacy for Diagnostics Analyzer looks for the file in /knowledgebase/ingested/.

“entitytype”

Specifies a name for the identifier. This is displayed in files generated by a REPORT request. If the **“inputsource”** of the identifier is **“csv”** or **“query”** in the ingestion_config.json file, the column name is to be used as **“entitytype”** and this specification is ignored.

For all other identifiers, if the entity name was provided when the file was ingested (for “custom” format), then this field can be skipped. If values are provided both during ingestion and in analysis_config.json, the value that is provided in analysis_config.json takes precedence. If no value is provided either during ingestion or in analysis_config.json, a custom value is chosen.

“description”

Specifies a description of the identifier. If values are provided both during ingestion and in analysis_config.json, the value that is provided in analysis_config.json takes precedence. If no value is provided either during ingestion or in analysis_config.json, a custom value is chosen.

“printable_characters”

This field specifies a set of printable characters, which is used for analysis of memory dumps. When analyzing the memory dump, only these characters are used to construct the tokens that are parsed. Default value is “ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890`~!@#%&^*()-=_+{}[];:'",<.>/?\\|\n ”

Any incorrect information in analysis_config.json is ignored. Identifiers that are specified in the **“built_in_identifiers_exclude”** list take precedence over identifiers that are specified in the **“built_in_identifiers_include”** list.


```

{
  "built_in_identifiers_include": ["Month", "FullName", "Credit Card Type",
  "Credit Card Number", "Email", "Zipcode", "Day"],
  "built_in_identifiers_exclude": ["Year", "Zipcode", "Date Time"],
  "custom_identifiers": [
    {
      "inputfilename": "acctnum.bin",
      "entitytype": "Account Number",
      "description": "List of account numbers",
      "format": "custom"
    },
    {
      "inputfilename": "policynum.bin",
      "entitytype": "PolicyNumber",
      "description": "List of policy numbers",
      "format": "custom"
    }
  ],
  "dependent_identifiers": [
    {
      "name": "Full Person",
      "identifiers": [ "FullName", "Zipcode", "Email" ]
    },
    {
      "name": "Card",
      "identifiers": [ "Credit Card Type", "Credit Card Number" ]
    }
  ],
  "built_in_ns_identifiers_include": ["ModuleName"],
  "built_in_ns_identifiers_exclude": [],
  "custom_ns_identifiers": [
    {
      "inputfilename": "branch.bin",
      "entitytype": "Branch Name",
      "description": "List of branch locations",
      "format": "custom"
    },
    {
      "inputfilename": "zone.bin",
      "entitytype": "Zone",
      "description": "List of zones",
      "format": "custom"
    }
  ]
}

```

Figure 46. *analysis_config.json* example

The "**built_in_identifiers**" supplied with Data Privacy for Diagnostics Analyzer are not necessarily all inclusive of the particular topic, and will not be changed over time even if there are changes to the actual data set of said topic. They are the following:

| Identifier (not case-sensitive) | Description |
|---------------------------------|---|
| age | String patterns that are related to age. Examples: "10 years old" "4 months old" "dob: 1-2-1999" Note: These string patterns are considered sensitive, but just the number 10 (in the first example) is not considered sensitive by itself. |
| continent | Dictionary containing the names of continents. |
| country | Dictionary containing the names of countries. |
| county | Dictionary containing the names of all the counties in the US. In the United States of America, an administrative or political subdivision of a state is a county. |

| Identifier (not case-sensitive) | Description |
|---------------------------------|---|
| credit card type | Credit card identification dictionary. Cards that are detected are VISA, Mastercard, AMEX, Diners Club, Discover, and JCB. |
| credit card number | Credit card pattern identification. Cards patterns that are detected are VISA, Mastercard, AMEX, Diners Club, Discover, and JCB. |
| date time | Date and Time pattern identification. |
| day | The dictionary containing the names of the days of the week. |
| dependent | The dictionary containing the names of types of dependents, such as daughter, son. |
| email | Email address pattern. |
| eu nin | National Identification Number patterns for various EU countries. |
| FirstName | A first name dictionary based on popular forenames in the US census. |
| FullName | <p>A given name and surname pair dictionary, the combination of which is from names in the US census.</p> <p>Note: This identifier detects a 2-word combination of a given name and surname that is separated by a delimiter of a comma, space, or both in either order. This identifier does not detect a name that contains any middle names, initials, hyphenated names, or names that contain apostrophes.</p> |
| gender | The dictionary that contains the genders Male and Female. |
| iban | International Bank Account Number (IBAN) pattern |
| icdv9 | International Classification of Diseases 9th Revision (ICDv9) identification dictionary. |
| icdv10 | International Classification of Diseases 10th Revision (ICDv10) identification dictionary. |
| imei | <p>International Mobile Equipment Identity (IMEI) identification dictionary.</p> <p>Note: This identifier only detects 15-digit IMEI tokens.</p> |
| imsi | International Mobile Subscriber Identity (IMSI) Identification dictionary. |
| in aadhaar card number | Aadhaar identification number for residents or passport holders of India. |
| in PAN card number | Permanent Account Number pattern issued by the Indian Income Tax Department |
| international phone number | International phone number identification pattern. |

| Identifier (not case-sensitive) | Description |
|---------------------------------|--|
| ip address | IP address identification pattern. Supports both IPv4 and IPv6 addresses |
| lastName | A last name dictionary based on popular surnames in the US census. |
| latitude longitude | Latitude and longitude identification pattern. Supports GPS and DMS coordinate formats, Ex: 12:30'23.256547S 12:30'23.256547E, N90.00.00 E180.00.00 |
| mac address | MAC Address Identification pattern. |
| marital status | Marital status identifier dictionary. |
| medical name | Medical Name identification pattern. Example: John Doe MD |
| medical record number | Medical Record Number identification pattern, for example, MRN: CLM-00000056055, Medical Record Number: 1234asds |
| month | Month Name identification dictionary. |
| occupation | Occupation identification dictionary. |
| PO box | Identifies post office box numbers, Ex: P.O. BOX 334, POBOX 14321412 |
| raceOrEthnicity | Dictionary identification of ethnic groups |
| religion | Dictionary containing major religions. |
| street types | Street Type identification, for example, tokens containing "st." |
| uk nin | National Insurance Number pattern. |
| us address | Identifies US-centric address patterns like "800 Theatre Court Garden City, NY 11530". This just checks the format but does not validate city, state, and ZIP code in the address. |
| us phone number | US-specific phone/fax/pager identifier pattern. |
| us ssn | US Social Security Number pattern |
| us states | US State Name identification |
| vehicle identification number | Vehicle identification number identification dictionary. Supports world manufacturer identification dictionary. |
| year | Year of Birth Identification, Any number between 0 and the current year. |
| zipcode | Valid US zip code identifier dictionary. |

Note: The following identifiers are no longer valid as of OA61591: Animal, ATC, Hospital Name, Phone Number, US SWIFT Code.

The list of **"built_in_ns_identifiers"** supplied with Data Privacy for Diagnostics Analyzer are:

The default `analysis_config.json` file includes the list of included identifiers and excluded identifiers.

Identifier checking is done by using the following order:

1. User feedback indicates that a token is sensitive.
2. User feedback indicates that a token is nonsensitive.
3. Module name check
4. Nonsensitive checks (built-in + custom)
5. Sensitive checks (built-in + custom)

Note: If the identifier is added into both the `_include` list and the `_exclude` list for either sensitive or nonsensitive checks, it is treated as excluded.

Requesting a REPORT run

REPORT is the function that formats concise reports in the file system into human-readable reports for the requested memory dump, and serves as the input to the FEEDBACK function. By default, the reports for the last ANALYZE run will be formatted. Regardless of how the job is initiated (by IPCS option 5.6 or by the BLSJDPR JCL), the runtime configuration file is used to specify options for processing. This file is either built by the dialog by using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPR JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPR JCL) are:

"input_dataset"

Specifies the location of the input memory dump. You must specify a data set name as follows: `"/<dump-dataset-name>".` During ANALYZE processing, the memory dump name was used as a directory and will be used by REPORT processing to locate the files produced during ANALYZE processing.

"dpfd_home"

Specifies the Data Privacy for Diagnostics Analyzer home directory.

"run_number"

Specifies the run number. If you omit the `run_number` option, it uses the most recent ANALYZE run.

As you validate how accurate the ANALYZE processing was in detecting sensitive data in your memory dumps, you might need to provide feedback to help the ANALYZE processing. In order to provide this feedback, you need to run the REPORT processing before the FEEDBACK function. The outputs of the report processing are human-readable files that can be edited should you need to provide FEEDBACK for future ANALYZE attempts. The files that are produced from the REPORT function are the following:

/<directory>/reports/<dump-name>/<run-number>/sensitive_tokens

This file contains the list of each token that was found to be sensitive data.

/<directory>/reports/<dump-name>/<run-number>/non_sensitive_tokens

This file, if requested, contains the list of each token that was found to be nonsensitive data. Note that this file is only produced if Page-Level Redaction was requested during the ANALYZE processing.

To ensure personal information and installation privacy requirements are protected, a thorough review of both the sensitive and non-sensitive token files is highly recommended.

Requesting a FEEDBACK run

After running the ANALYZE processing, followed by the REPORT processing, you might want to provide feedback to enhance the accuracy of future ANALYZE functions in detecting the appropriate sensitive data for your environment. You can provide feedback to indicate the following:

- Tokens found to be sensitive that are not sensitive.
- Tokens not found to be sensitive that are sensitive.

To do this, you must edit the reports that are generated from the REPORT processing. These reports are found in the `/<directory>/reports/<dump-name>/<run-number>` directory.

- In the `sensitive_tokens` file, change the `"Is_Analysis_Correct"` field from `"Y"` to `"N"` for any token that should not be considered sensitive.

- In the `non_sensitive_tokens` file, change the “Is_Analysis_Correct” field from “Y” to “N” for any token that should be considered sensitive.

Afterward, the FEEDBACK function can be requested. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPF JCL), the runtime configuration file is used to specify options for processing. This file is either built by the dialog by using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPF JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPF JCL) are:

"input_dataset"

Specifies the location of the input memory dump. You must specify a data set name as follows: `///<dump-dataset-name>'`. During ANALYZE processing, the memory dump name was used as a directory and will be used by FEEDBACK processing to locate the files produced during ANALYZE processing.

"dpfd_home"

Specifies the Data Privacy for Diagnostics Analyzer home directory.

"run_number"

Specifies the run number. If you omit the `run_number` option, it uses the most recent ANALYZE run. Ensure that the `run_number` is the same directory in which the edited reports are contained.

Note: If the redaction string is marked as a sensitive token via FEEDBACK, it will not be treated as a sensitive token.

During the FEEDBACK operation, the Data Privacy for Diagnostics Analyzer reads these edited reports and updates the `<directory>/knowledgebase/feedback/feedback.bin` file, which will be used for future ANALYZE runs.

Requesting an INGEST run

You might want to customize the detection of sensitive data that is unique to your environment. This customization can be achieved by the INGEST function, which will help the Data Privacy for Diagnostics Analyzer to detect the sensitive data in future analysis. You can initiate the INGEST function from either IPCS option 5.6 or the BLSJDPI JCL. Either way, it uses the following files:

Runtime configuration file

This file is either built by the dialog by using the supplied parameters, or is specified by an in-stream DD statement in the BLSJDPI JCL.

ingestion_config.json file

This file provides detail on what the Analyzer, which then can be used while analyzing diagnostic data. It is located in the `directory/configuration/` directory in the Data Privacy for Diagnostics file system. You can use either the **EDIT CONFIG FILE** option Y in the **INGEST IPCS** panel to edit this file if you want to change it, or you can directly edit it using an editor that you are familiar with.

Regardless of how the job is initiated, the runtime configuration file is used to specify options for processing. The runtime configuration file parameters (Built by dialog or hand-coded in the BLSJDPI JCL) are as follows:

"dpfd_home"

Specifies the Data Privacy for Diagnostics Analyzer home directory.

The `ingestion_config.json` file contains information about the identifiers to be built as sensitive or nonsensitive tokens based on the options specified. INGEST generates a file in the `directory/knowledgebase/ingested` directory, which can be then specified in the `analysis_config.json` file to add the ingested identifier as a custom identifier for sensitive data detection during the ANALYZE function. This data can be ingested from dictionaries, databases, or other sources.

"outputfilename"

Specifies the name of the file to be stored in the `directory/knowledgebase/ingested` directory after successful INGEST run. This parameter can be specified in the `analysis_config.json` file as the **"inputfilename"** under the **"custom_identifiers"** option for use in future ANALYZE requests.

"entitytype"

Specifies the name of the identifier. This parameter is used in future REPORTs when a user does not provide one in the `analysis_config.json` file in the **"custom_identifiers"** option under **"entitytype"**. If the **"inputsource"** of the identifier is **"csv"** or **"query"** in the `ingestion_config.json` file, the column name is to be used as **"entitytype"** and this specification is ignored.

"description"

Specifies a description of the identifier. This parameter is used when user does not provide one in the `analysis_config.json` file in the **"custom_identifiers"** option under **"description"**. If the **"inputsource"** of the identifier is **query**, then the description must contain a list of strings with one description per query.

"inputtype"

Specifies the type of input to INGEST. The following values are valid:

"pattern"

Specifies Java Regex (or Java Regular Expression) as a pattern for matching strings.

"dictionary"

Specifies exact tokens the Data Privacy for Diagnostics Analyzer matches.

"inputsource"

Specifies the source of the input data to be ingested. Valid values are:

"file"

Indicates that the source is a file with the location that is specified on the **"inputfilename"** option.

"inline"

Indicates that the source is inline data that is specified in the `ingestion_config.json` file.

"database"

Indicates that the source is a database as specified in the **"database"** and associated options. This option is only valid when **"inputtype"** of **"dictionary"** is specified.

Note: If you chose this option, you must update the DB2 JDBC PATH, by either entering the full path to the Db2 JDBC Driver and License JARs in the Db2 JDBC PATH field in IPCS option 5.6 or update the CLASSPATH section of the STDENV DD in the BLSJDPI JCL to include the full path.

"csv"

Indicates that the source is a CSV file with the location that is specified on the **"inputfilename"** option. When choosing this option, the first row is treated as the column name or header for the data that is contained within. The result of a CSV ingest operation is one aggregate proxy identifier that contains the entirety of the CSV file and an individual identifier for each column that is contained within. This option is only valid when **"inputtype"** of **"dictionary"** is specified.

"query"

Indicates that the source is a list of SQL queries issued against the database as specified in the **"database"** and associated options. The result of a query ingest operation is one aggregate proxy identifier that contains the entirety of the resultant list of queries and an individual identifier for each column that is contained in each of the query results. This option is only valid when **"inputtype"** of **"dictionary"** is specified.

Note: If you chose this option, you must update the DB2 JDBC PATH, by either entering the full path to the Db2 JDBC Driver and License JARs in the DB2 JDBC PATH field in IPCS option 5.6 or update the CLASSPATH section of the STDENV DD in the BLSJDPI JCL to include the full path.

"outputfilename"

Specifies the name of the file to be stored in the `directory/knowledgebase/ingested` directory after a successful INGEST run. This parameter can be specified in the `analysis_config.json` file as the **"inputfilename"** under the **"custom_identifiers"** option for use in future ANALYZE requests. If the **"inputsource"** of the identifier is **"query"**, then the **"outputfilename"** must contain a list of files with one file per query for output. For **"inputsource"** of **"csv"** or **"query"**, there may be multiple files in the `directory/knowledgebase/ingested` directory due to the

nature of using the aggregate proxy identifier plus the column delineated individual identifiers. These column delineated files start with the one or more names that are listed in the **"outputfilename"** field and is appended with a "-" followed by the individual column name, ending with the file suffix specified.

"inputfilename"

Specifies the path and name of the file, which contains the data to be used during the INGEST function. This option is only valid when **"inputsource": "file"** or **"csv"** is specified.

For **"inputsource": "file"**, the format for specifying a dictionary or pattern in a file is one entry per line as follows:

```
value1
value2
value3
```

For **"inputsource": "csv"**, the csv file must contain a header row as the first row, which is to be used as the column names for **"entitytype"** and the rest of the rows contain the data to be ingested.

"inlinedata"

Specifies the data to be used during the INGEST function. This option is only valid when **"inputsource": "inline"** is specified. The format for specifying inline data is: **"inlinedata"** : ["value1", "value2", "value3"]

"database"

Specifies the type of database to be used as an input source. This option is only valid when **"inputsource"** of **"database"** or **"query"** is specified. Valid values are:

"DB2"

Indicates that the database is Db2 installed on nonsystem Z platform.

"DB2zOS"

Indicates that the database is Db2 installed on IBM Z®.

"DB2zOSptkt"

Indicates that the database is Db2 to be connected by PassTicket. This is the default value.

"databasehost"

Specifies the domain name or IP address where the database is hosted. This option is only valid when **"inputsource"** of **"database"** or **"query"** is specified. The format for specifying this option is:

"databasehost": "url"

"databaseport"

Specifies the port number that identifies the Db2 subsystem. This option is only valid when **"inputsource": database** or **"query"** is specified.

"databaseusername"

Specifies the user ID used to connect to the Db2 database. This option is only valid when **"inputsource": database** or **"query"** is specified.

"databasepassword"

Specifies the password for the user ID used to connect to the Db2 database. This option is only valid when **"inputsource": database** or **"query"** is specified.

"databaselocation"

Specifies the name of the database that contains the data to be ingested. This option is only valid when **"inputsource": database** or **"query"** is specified.

"databasename"

Specifies the name of the database that contains the data to be ingested. This option is only valid when **"inputsource": "database"** is specified.

"databaseschema"

Specifies the schema in the database that contains the data to be ingested. This option is only valid when **"inputsource": "database"** is specified.

“databasetablename”

Specifies the name of the table in the database that contains the data to be ingested. This option is only valid when **“inputsource”:“database”** is specified.

“databasecolumnname”

Specifies the name of the column in the database that contains the data to be ingested. This option is only valid when **“inputsource”:“database”** is specified.

“database query”

Specifies a list of queries to be issued against the database and the results are ingested. This option is only valid when **“inputsource”:“query”** is specified.

Any options specified in the `ingestion_config.json` file that are not valid with the specified input source are ignored. After the INGEST request is completed, the newly created identifier must be specified in the **“custom_identifiers”** options of the `analysis_config.json` file to be considered for determining sensitive data for the subsequent ANALYZE requests.

Note: Only 1 custom identifier may be specified per INGEST request for **“inputsource”** of file, inline, or database. If more than 1 identifier is wanted per INGEST request, use **“csv”** or **“query”** as **“inputsource”**.

The following are examples of `ingestion_config.json` files.

Inline Pattern Example

To create a new sensitive identifier called account that detects the account number of customers in a memory dump, a pattern can be used if a certain format for the account numbers is known. For example, an account number beginning with two alphabetical characters followed by eight numeric digits. The following is an inline pattern by using Java Regex.

```
{
  "inputtype": "pattern",
  "inputsource": "inline",
  "entitytype": "account",
  "description": "a pattern to determine account: 2 characters, 8 digits",
  "outputfilename": "accts.bin",

  "inlinedata": ["\\D{2}\\d{8}"]
}
```

Note: \ is an escape character in Java strings, which requires you to use \\ to define a single \ to use meta Regex characters like \D (nondigit) and \d (digit).

File Dictionary Example

If you have a file that contains all the account numbers for your customers, you can alternatively provide a file dictionary to create your custom identifier:

```
{
  "inputtype": "dictionary",
  "inputsource": "file",
  "entitytype": "accounts",
  "description": "a list of our customer accounts",
  "outputfilename": "accts.bin",

  "inputfilename": "/u/ibmuser/accounts.txt"
}
```

Db2 Database Dictionary Example

You can also INGEST a column from a Db2 database as a dictionary that can be used to identify sensitive data:

```
{
  "inputtype": "dictionary",
  "inputsource": "database",
  "entitytype": "accounts",
  "description": "a list of our customer accounts",
```



```
"outputfilename": "accts.bin",
"database": "db2zos",
"databasehost": "db2host.ibm.com",
"databaseport": "446",
"databaseusername": "db2user",
"databasepassword": "db2pw",
"databaseselocation": "DB2LOC",
"databaseschema": "dsn8910",
"databasetablename": "ACCTS",
"databasecolumnname": "ACCOUNTNUMBER"
}
```

After the INGEST process is completed, to use your newly created customer identifier in the subsequent ANALYZE request, you must amend your `analysis_config.json` file to add the accounts to the custom identifier as such:

```
"custom_identifiers":
[
    {
        "inputfilename" : "accts.bin",
        "entitytype" : "accounts",
        "description" : "Customer Accounts",
        "format" : "custom"
    }
]
```

CSV Dictionary Example

To import multiple fields of information in a single operation, INGEST with the csv as **"inputsource"** can be used if customer information is in a csv file:

```
{
  "inputtype": "dictionary",
  "inputsource": "csv",
  "entitytype": "clients",
  "description": "names and emails of clients",
  "outputfilename": "clients.bin",
  "inputfilename": "/u/ibmuser/clients.csv"
}
```

If the `clients.csv` file contains 3-column headers FIRSTNME, LASTNAME, EMAILADR with subsequent data in other rows, then the output in `directory/knowledgebase/ingested` contains 4 files, the aggregate proxy identifier, and the 3 individual column identifiers:

```
clients.bin
clients-EMAILADR.bin
clients-FIRSTNME.bin
clients-LASTNAME.bin
```

The `clients.bin` file can be used in the `analysis_config.json` file to add it as a custom identifier that contains the entire CSV file:

```
"custom_identifiers":
[
    {
        "inputfilename" : "clients.bin",
        "entitytype" : "ourclients",
        "description" : "names and emails of clients",
        "format" : "custom"
    }
]
```

This is equivalent to specifying all the individual column identifiers:

```
"custom_identifiers":
[
    {
        "inputfilename" : "clients-EMAILADR.bin",
        "entitytype" : "emailadr",
        "description" : " emails of clients",
        "format" : "custom"
    },
    {
        "inputfilename" : "clients-FIRSTNME.bin",
        "entitytype" : "firstnme",
        "description" : "first names of clients",
        "format" : "custom"
    },
    {
        "inputfilename" : "clients-LASTNAME.bin",
        "entitytype" : "lastname",
        "description" : "last names of clients",
        "format" : "custom"
    }
]
```

```

        "inputfilename" : "clients-FIRSTNME.bin",
        "entitytype" : "firstnme",
        "description" : " first names of clients",
        "format" : "custom"
    },
    {
        "inputfilename" : "clients-LASTNAME.bin",
        "entitytype" : "lastname",
        "description" : " last names of clients",
        "format" : "custom"
    }
]

```

One can specify an individual column file if wanted by using the above example.

Db2 Query Dictionary Example

To import multiple fields of information from Db2 in a single operation, use INGEST with the query **"inputsource"**:

```

{
  "inputtype": "dictionary",
  "inputsource": "query",
  "entitytype": "mysql",
  "description": ["S1T1", "S2T1", "S1T3"],
  "outputfilename": ["s1t1.bin", "s2t1.bin", "s1t3.bin"],
  "database": "db2zos",
  "databasehost": "db2host.ibm.com",
  "databaseport": "446",
  "databaseusername": "db2user",
  "databasepassword": "db2pw",
  "databaselocation": "DB2LOC",
  "databasequery": ["SELECT FIRSTNME AS F1, LASTNAME AS L1 FROM SCHEMA1.TABLE1",
    "SELECT EMAILADR FROM SCHEMA2.TABLE1",
    "SELECT * FROM SCHEMA1.TABLE3"]
}

```

This ingests 3 queries, hence the list of 3 strings in **"description"** and 3 files in **"outputfilename"** fields. The first query generates an aggregate proxy identifier that is named *s1t1.bin* with the other 2 columns that are named *s1t1-F1.bin* and *s1t1-L1.bin* since the columns were renamed by using the AS keyword in SQL. The second query generates an aggregate proxy identifier that is named *s2t1.bin* with a column identifier of *s2t1-EMAILADR.bin*. The third query results in n+1 identifiers with an aggregate proxy identifier named *s1t3.bin* and the individual column identifiers start with *s1t3-* and end with the column name followed by the *.bin* suffix due to the wild carding selecting all columns from *SCHEMA1.TABLE3*.

The aggregate proxy files and individual column files can be used in a similar manner to those in the csv **"inputsource"** example above.

Requesting an EXTRACT run

You may want to know what criteria that ANALYZE is using to determine the sensitivity of data.

This can be achieved by the EXTRACT function, which writes the pattern or dictionary of an identifier (built-in or custom) to a file. You can initiate the EXTRACT function from either IPCS option 5.6 or the BLSJDPX JCL. Either way, it uses the following files:

Runtime Configuration File

This file is either built by the dialog by using the supplied parameters, or is specified by an in-stream DD statement in the BLSJDPX JCL.

extract_config.json file

This file provides additional detail on what to include or exclude while looking for sensitive data in memory dump records. It is located in the *home/configuration/* directory in the Data Privacy for Diagnostics file system. You may either use the **EDIT CONFIG FILE** option Y in the **EXTRACT IPCS** panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

Regardless of how the job is initiated, the runtime configuration file is used to specify options for processing. The runtime configuration file parameters (Built by dialog or hand-coded in the BLSJDPX JCL) are as follows:

"dpfd_home"

Specifies the Data Privacy for Diagnostics home directory.

The `extract_config.json` file contains information about the identifiers to be written to a file based on the options specified. EXTRACT generates a file in the specified output directory. The identifiers can be either built-in or custom from a prior INGEST operation.

Parameter Descriptions:

"identifiers_list"

Specifies the array of identifiers to be written to a file. Each identifier should have a different output file as to not over-write the previous identifier. The format for specifying an **"identifiers_list"** is:

```
"identifiers_list":[
{
  "identifier_type": "<type1>",
  "identifier_name": "<name1>",
  "output_filename": "<outputpath1>"
},
{
  "identifier_type": "<type2>",
  "identifier_name": "<name2>",
  "output_filename": "<outputpath2>"
}
]
```

"identifier_type"

Specifies the type of the identifier. Valid values are **"builtin"** for built-in identifier or **"custom"** for a custom identifier from an INGEST request.

"identifier_name"

Specifies the name of the identifier.

- If a built-in identifier is specified, this is the name of the built-in identifier as defined in [Figure 45 on page 64](#)
- If a custom identifier is specified, this is the file name in the knowledgebase/ingested directory.
- If an aggregate proxy identifier is specified, EXTRACT outputs the individual column identifiers for this aggregate proxy identifier.

"output_filename"

Specifies the full path for the file where the extracted information is written. If using an aggregate proxy identifier as the **"identifier_type"**, extract will append the individual column name after the file name specified and then append the suffix that are specified. For example, if the **"output_filename"** field is `/DPfD/mycsv.out` and this CSV file has 2 columns that are named `COL1` and `COL2`, the output from the extract operation is 2 files that are named `/DPfD/mycsv-COL1.out` and `/DPfD/mycsv-COL2.out` containing the contents of each column respectively.

extract_config.json Example

If you want to extract both a built-in identifier (person) and custom identifier (the accounts identifier that were previously defined in the INGEST request section), then your **"identifiers_list"** would be the following:

```
"identifiers_list":[
{
  "identifier_type": "builtin",
  "identifier_name": "FullName",
  "output_filename": "/DPfD/names.txt"
},
{
  "identifier_type": "custom",
  "identifier_name": "accts.bin",
  "output_filename": "/DPfD/accounts.txt"
}
```

Chapter 4. Using the User and Sysplex Dump Directories

A dump directory is a VSAM data set that contains information about unformatted sources that IPCS can format, such as: an SVC dump, a stand-alone dump, an SYSMDUMP dump, a trace data set, a data set, or active storage. The two kinds of dump directory are:

- A user dump directory, which each IPCS user must have to help IPCS format and analyze sources.
- A sysplex dump directory, which should be shared by systems in a sysplex. Each system automatically adds information to the sysplex dump directory about every SVC dump it writes.

A dump directory contains *source descriptions*, which describe a source and consist of:

- **Symbol Table:** A table of symbols that represent control blocks and data areas for a source. The symbol table lets you use a label to refer to a location in a dump, rather than an address. See [“Using the Symbol Table” on page 90](#).
- **Storage Map:** A mapping of the storage for a source. The storage map shows which control blocks, modules, and storage areas are in a section of storage in the dump. See [“Using the Storage Map” on page 94](#).
- **Other Records:** Other records used by IPCS to describe dump and trace data sets. The other records contain, among other things, the dump title, symptoms, and other key serviceability data.
- **Local and Global Defaults:** The defaults in effect for a user. The default lists allow IPCS to retain your defaults between IPCS sessions. The defaults appear only in a user dump directory.

This chapter describes dump directories in the following topics:

- [“Using a User Dump Directory” on page 83](#)
- [“Using the Sysplex Dump Directory” on page 83](#)
- [“Managing Dump Directories” on page 84](#)
- [“Changing Your Current User Dump Directory” on page 96](#)
- [“Administering the Sysplex Dump Directory” on page 97](#)

Using a User Dump Directory

Each IPCS user has a user dump directory. For you to format a dump or trace, your current user dump directory must contain a source description for it. If the directory does not, IPCS initializes the dump or trace and creates a source description for it in your current user dump directory.

The user dump directory is allocated when you access IPCS. See [Chapter 2, “Starting IPCS,” on page 13](#) for more information about how this is done.

IPCS records the defaults in your user dump directory when you set them and reinstates the defaults at the beginning of your next IPCS session.

You can use only one dump directory at a time during an IPCS session. However, you can define a different user dump directory, for example, for each dump or group of dumps. Multiple dump directories allow you to use one directory when processing problems in one group and another directory when processing problems in another group. For ease in assigning and reassigning problems, you might want a separate dump directory for each problem.

Using the Sysplex Dump Directory

The sysplex dump directory is an optional VSAM data set that describes the SVC dumps generated by a sysplex in a central, compact, and manageable place. If you have write access, you can add source descriptions for other unformatted dumps that IPCS can format and for trace data sets.

When setting up the sysplex dump directory, arrange for all systems in the sysplex to share it:

- Use the default name of SYS1.DDIR for the sysplex dump directory or specify the same name for it in the SYSDDIR statement in the BLSCUSER parmlib member.
- Place the data set for the sysplex dump directory on a DASD shared by all systems in the sysplex.

When setting up the sysplex dump directory on a running sysplex for the first time, follow the listed considerations:

- Use the BLSCDDIR CLIST to allocate the dump directory data set on the first system.
- Ensure that the same name is in the SYSDDIR statement in the BLSCUSER PARMLIB member for every system in the sysplex.
- Ensure that the DASD, that the data set is on, is available to all of the systems.
- Ensure that the data set is cataloged as being in the same place for every system in the sysplex; especially if some systems are using different catalogs.
- Issue 'S BLSJPRMI' on every system to make sure that IPCS will use the updated BLSCUSER member.
- While it might be excessive if a dump has not already occurred (and you will lose any captured dumps that are not written out to DASD, if resources are not available), issue 'CANCEL DUMPSRV' to force SDUMPX processing to use the (potentially) new sysplex dump directory.

When a system that has access to a sysplex dump directory generates an SVC dump, the system automatically records the source description for it in the sysplex dump directory. IPCS adds the source description without initializing the dump, which takes time. If possible, IPCS accesses the dump to define symbols for it and place them in the symbol table in the source description.

IPCS sets up the source descriptions for dumps and traces in the same way for both the sysplex and the user dump directories. As a consequence, you can transfer a source description of a dump or trace from the sysplex dump directory to your user dump directory, so that you can analyze the dump or trace.

Authorized users can access the sysplex dump directory and edit it. You obtain authorization through the z/OS Security Server, which includes RACF®.

Managing Dump Directories

Managing a user or sysplex dump directory is described in the topics:

- [“Preparing a Dump Directory” on page 84](#)
- [“Maintaining a Dump Directory” on page 87](#)
- [“Changing the Characteristics of a Dump Directory” on page 89](#)
- [“Using the Symbol Table” on page 90](#)
- [“Using the Storage Map” on page 94](#)

Preparing a Dump Directory

You can prepare a dump directory in one of two ways:

- Using the BLSCDDIR CLIST. For a new IPCS user who does not have a user dump directory, the TSO/E IPCS command invokes the BLSCDDIR CLIST to prepare a directory.
- Using a three-step process, which allows you to customize the directory.

Using the BLSCDDIR CLIST

IPCS supplies the BLSCDDIR CLIST in the SYS1.SBLSCLI0 system data set to create or modify a dump directory. Use this CLIST to:

- Create a user dump directory when accessing IPCS for the first time
- Create user dump directories to satisfy special needs

- Create multiple user dump directories — for example, to use for simultaneous interactive and batch processing
- Create a sysplex dump directory

When you start an IPCS session but do not have a user dump directory, the TSO/E IPCS command invokes BLSCDDIR automatically. BLSCDDIR uses installation-defined defaults for data set name, size, volume, and associated ddname name. The installation can customize BLSCDDIR to control the size and volume defaults; see [z/OS MVS IPCS Customization](#).

You can create a user dump directory for your own special uses by specifying parameters on the BLSCDDIR CLIST invocation.

Creating a Sysplex Dump Directory

To create a sysplex dump directory with the default name of SYS1.DDIR, enter the following, for example:

```
%blscddir dsn(sys1.ddir)
```

This example assumes:

- You have access to SYS1.SBLSCLI0 through the SYSPROC DD statement or by the TSO/E ALTLIB command.
- You want to place the sysplex dump directory on a volume named VSAM01.
- You want to use the default name of SYS1.DDIR. If not, specify the name in the SYSDDIR statement in the BLSCUSER parmlib member for all systems in the sysplex.

See the BLSCDDIR CLIST in [z/OS MVS IPCS Commands](#) for more information.

Using a Three-Step Process

If you want to customize the directory, use the following three steps to create it:

1. Create a VSAM data set for the dump directory.
2. Initialize the VSAM data set as a dump directory.
3. Allocate the VSAM data set as a dump directory.

Creating a VSAM Data Set for the Dump Directory

Use the access method services DEFINE command to create the VSAM data set:

```
DEFINE CLUSTER(NAME('userid.DDIR') VOLUMES(VSAMnn) REUSE)
  INDEX(NAME('userid.DDIR.I') TRACKS(1 1) )
  DATA(NAME('userid.DDIR.D') CYLINDERS(1 1) KEYS(128 0)
    RECORDSIZE(384 3072) )
```

The DEFINE command is described in [DEFINE CLUSTER](#) in [z/OS DFSMS Access Method Services Commands](#)

If the creation is successful, the system displays the following messages:

```
IDC0508I DATA ALLOCATION STATUS FOR VOLUME VSAMnn IS 0
IDC0509I INDEX ALLOCATION STATUS FOR VOLUME VSAMnn IS 0
READY
```

The DEFINE command creates three data sets:

- *userid.DDIR*
- *userid.DDIR.I*
- *userid.DDIR.D*

The dump directory is a VSAM key-sequenced cluster. If an extended format data set is used, you may request compression, especially in a system running stable production loads that produce few dumps

and, thus, require few updates to the dump directory. However, compression might save little space because much of the data is placed in the keys, which are not compressed.

The dump directory has the characteristic SHAREOPTIONS(1). Data sets with SHAREOPTIONS(1) cannot be open for concurrent update by multiple users (or a user and that user's own batch jobs). Only one IPCS session can use a dump directory at a given time.

Determining the Appropriate Size for a Dump Directory

The DASD space required depends on the number of dumps described in the directory, the size of each dump, and the amount of information for each dump. Suggested sizes are:

- For a user dump directory, allocate the number of tracks equal to 4 to 6% of the total number of tracks in the dumps to be in the directory. If you use symbol tables and storage maps extensively, your user dump directory will need more space.
- For a sysplex dump directory, allocate at least 5000 bytes of space for each dump that may be described in the directory. The number of dumps depends on how often an administrator will remove old entries.

Note:

1. You can replace *userid.DDIR* with data set names meaningful to you if you prefer.
2. Use a continuation character for each line as appropriate.
3. Check with your installation data base administrator regarding the specification of VSAM volumes. To promote consistent practices within your installation, you might be asked to use an installation-supplied CLIST that creates and initializes your dump directory. See [“Using the BLSCDDIR CLIST” on page 84.](#)

Initializing the VSAM Data Set as a Dump Directory

After you create a new dump directory VSAM data set, you must initialize it for IPCS. Use the TSO/E IPCSDDIR command as follows:

```
IPCSDDIR 'userid.DDIR'
```

Note: You do not need to initialize your dump directory for each IPCS session.

If the initialization is successful, the system displays READY.

The IPCSDDIR command writes two records to the dump directory: one with a key of binary zeros (0) and the other with a key of binary ones (1).

Allocating the VSAM Data Set as a Dump Directory

You must allocate a dump directory for each IPCS session. IPCSDDIR is the standard ddname for the IPCS dump directory. IPCS has provisions to ensure the data set's integrity.

If you do not want to let the TSO/E IPCS command allocate the dump directory, you must allocate a dump directory before entering IPCS. The TSO/E ALLOCATE command allocates the data set *userid.DDIR* to the ddname IPCSDDIR:

```
ALLOCATE DDNAME(IPCSDDIR) DSNAME('userid.DDIR') SHR
```

When you start an IPCS session, IPCS opens the data set allocated to IPCSDDIR. If you did not allocate the data set — whether by entering the ALLOCATE command or by using a CLIST — or if IPCS cannot open it, the TSO/E IPCS command ends.

Do not allocate IPCSDDIR to the batch local shared resources (LSR) subsystem. IPCSDDIR must be allocated to a data set.

Maintaining a Dump Directory

As the dump directory fills up, it might be necessary to delete symbols and storage map entries in order to free space. Use the following subcommands to maintain a dump directory:

| When You Want to | Use this Subcommand |
|---|---------------------|
| Copy source descriptions from one dump directory to the current dump directory | COPYDDIR |
| Delete all records that describe a particular dump or trace from the dump directory | DROPDUMP |
| List dumps and traces represented in the dump directory | LISTDUMP |

You can also use the TSO/E REPRO command to refresh a fragmented dump directory.

Processing Sources Both in Batch and Interactively

If you ask IPCS to use a dump directory that is being used by another IPCS session, IPCS responds with message BLS21101I to suggest you try again later.

If you submit a background job that initiates an IPCS session, and another IPCS session is using the dump directory required by the background job, IPCS waits for the dump directory to become available to process the job.

If you want to process a dump or trace interactively without tying up or waiting for the dump directory from the batch IPCS session, you must use a different dump directory.

Copying Source Descriptions to Your User Dump Directory

Use the following procedure to copy a source description for a dump from the sysplex dump directory or another dump directory, perhaps the one used for a batch IPCS session. In this example, SYS1.DDIR is the name of the sysplex dump directory.

- Choose the UTILITY option on the IPCS Primary Option Menu, then the COPYDDIR option on the IPCS Utility Menu.
- Specify the source data set name. Note that you need READ access to the sysplex dump directory.

```
----- Copy Dump Directory Data -----  
COMMAND  ===>  
  
Enter or verify the name of the source dump directory. Use ENTER to view the  
inventory for that directory, END to terminate.  
  
Source dump directory ===> 'SYS1.DDIR'
```

- Press Enter. IPCS displays the following output:

```

IPCS COPYDDIR from SYS1.DDIR -----
                        to IPCS1.DDIR

Command ==> ----- SCROLL ==> CSR_

AC Dump Source                                         Status
__ DSNAME('ILM.TSTDUMP4') . . . . .
CLOSED
    Title=TSTDMP4
    No symptoms
__ DSNAME('IPCS1.JBB7713.SYS1.D000601') . . . . .
CLOSED
    Title=COMPON=COMMTASK, COMPID=SC1CK, ISSUER=IEAVMFRR-FRR, COMMUNICATIONS TASK
    Psym=RIDS/IEECVSCR#L RIDS/IEECVSCR PEDS/5752SC1C4 AB/S00C1 RIDS/IEAVMFRR#R R

```

The panel identifies the source and target directories and displays the data sets described by the source directory. To copy one or more of these descriptions to your personal directory, enter selection code CO to the left of the entry. The complete list of selection codes that can be chosen are:

```

CL
  CLOSE

CO
  COPYDDIR

DD
  DROPDUMP RECORDS(ALL)

LA
  LISTDUMP SELECT(ATTRIBUTES)
  INDSNAME(source-directory)

LB
  LISTDUMP SELECT(BACKING)
  INDSNAME(source-directory)

LD
  LISTDUMP SELECT(DUMPED)
  INDSNAME(source-directory)

LT
  LISTDUMP SELECT(TRANSLATION)
  INDSNAME(source-directory)

LZ
  LISTDUMP SELECT(all-the-above)
  INDSNAME(source-directory)

SD
  SETDEF

```

- Press PF3 to return to the Copy Dump Directory Data panel, which tells you that the copy function is complete:

```

----- Copy Dump Directory Data ----- COPY COMPLETE
COMMAND  ===>

Enter or verify the parameters to copy data to your current dump directory.
Use ENTER to copy data, END to terminate.

Copy these DUMP DESCRIPTIONS (specify one or both of the following):
  DSNAMES ===> 'D46IPCS.DUMP.SVCVER4'
  DDNAMES ===>

From this DUMP DIRECTORY ===> 'SYS1.DDIR'

```

Dump Directories and Performance

IPCS relies on a dump directory for efficient processing of dumps. After a period of use, however, the dump descriptions in these VSAM data sets become fragmented.

Changing the Control Interval Size

Increasing the control interval size of the dump directory helps decrease the number of splits during dump processing. See [STA: Statistics Group](#) in *z/OS DFSMS Access Method Services Commands* for information about control interval splits. For information about changing the IPCS-supplied default in BLSCDDIR, see [Editing the BLSCDDIR CLIST](#) in *z/OS MVS IPCS Customization*.

Refreshing a Dump Directory

The following figure is a sample CLIST to reorganize a dump directory. It copies the dump descriptions into a temporary data set, then copies them back into the dump directory, allowing VSAM to rearrange the data.

```

PROC 1 DSN
CONTROL LIST
ALLOCATE FILE(SAVEDDIR) UNIT(SYSVIO) CYL SPACE(10 10) RECFM(U) +
BLKSIZE(3072) REUSE NEW
REPRO INDATASET(&DSN) OUTFILE(SAVEDDIR)
REPRO INFILE(SAVEDDIR) OUTDATASET(&DSN) REUSE
FREE FILE(SAVEDDIR)

```

Figure 47. Sample CLIST to Refresh a Dump Directory

Using the Sysplex Dump Directory for Initialization

When IPCS initializes a dump, it writes a large amount of information about a dump to a dump directory. If you have WRITE access to the sysplex dump directory, use it for the initial dump analysis, then copy the description into your user dump directory using the IPCS COPYDDIR subcommand. COPYDDIR allows VSAM to rearrange the dump description as it is copied to your dump directory.

If your system is not part of a sysplex, you can achieve the same effect by using a central dump directory for the initialization.

Changing the Characteristics of a Dump Directory

If during IPCS processing you find that you require dump directory attributes that are different than those defined by your installation, follow this procedure:

1. Use the TSO/E REPRO command to save the contents of your user dump directory in another data set.
2. Delete the unsatisfactory user dump directory data set.
3. Invoke the BLSCDDIR CLIST to create, initialize, and allocate a user dump directory more suitable to your needs, using the appropriate parameters to specify a different name, more space, or a different volume.
4. Use the TSO/E REPRO command to restore the contents of the user dump directory.

Using the Symbol Table

For every dump you process with IPCS subcommands, IPCS maintains a table of symbols representing data areas in the dump. The symbol table is in the source description for the dump your user dump directory. IPCS uses the table to make future references to a dump faster and more efficient.

When you process a dump for the first time, IPCS creates a source description for the dump in your user dump directory. IPCS reads the dump and initializes the symbol table in the source description. A symbol in IPCS is a label associated with a location in a dump. IPCS automatically creates some symbols for you. You can create your own symbols for areas of the dump using the EQUATE subcommand.

The symbol table is a list of symbol names, dump addresses, and symbol attributes. The attributes can include any one of the data description parameters, such as an address space identifier (ASID), CPU, and length; see [Chapter 5, “Describing Storage in a Dump,” on page 99](#). The symbol can also have the DROP or NODROP attribute.

Symbols Created by IPCS

IPCS uses the symbol X to represent the current address in the current dump. X is automatically given the NODROP attribute. You can specify X as an address parameter to many IPCS subcommands. Various IPCS subcommands might modify X as a result of their processing.

Many IPCS subcommands define symbols as part of their output. For example, to ask IPCS to find and display the ASVT wherever it is, specify:

```
list asvt structure(asvt)
```

If the symbol ASVT is not already in the symbol table, LIST locates it, verifies it, enters it in the symbol table with the attribute of STRUCTURE(ASVT), and lists it.

However, LIST displays the storage at the location pointed to from X'5820' if you specify:

```
list 5820.% structure(asvt)
```

The subcommand does not know if this really is the ASVT. It does not create a symbol table entry for ASVT, nor does it use such an entry if it already exists. In this example, you are telling the subcommand that the storage at the specified location is the ASVT. The subcommand verifies that the storage is the ASVT but does not make a symbol table entry for the ASVT.

When processing a stand-alone dump, IPCS analysis subcommands might have to simulate dynamic address translation (DAT). There are many control blocks that the subcommands use to do this translation. An IPCS subcommand that accepts a data description parameter may locate and validate the control blocks it needs to perform its function (if they are not already located) and make entries for them in the symbol table and storage map.

The FINDMOD, FINDUCB, RUNCHAIN, SCAN, and SUMMARY subcommands locate modules and control blocks in the current dump. FINDMOD locates a module, FINDUCB locates a UCB, RUNCHAIN processes a specified chain of control blocks, SUMMARY locates ASCBs. As each subcommand finds what it is looking for, it generates a name for it and puts that name in the symbol table along with the attributes of the module or control block it represents.

IPCS also adds symbols to the table when processing the CTRACE and GTFTRACE subcommands. Component buffer find routines produce symbols that describe buffers in a dump when processing the CTRACE subcommand. Processing for GTFTRACE adds a symbol that notes the wrap point of a wrapped trace data set.

The symbols created by IPCS are named according to a naming convention (see [“Naming Conventions for Special Symbols” on page 91](#)). You can refer to them by name once IPCS processes the subcommand that defines them.

Naming Conventions for Special Symbols

When IPCS creates a symbol definition, it uses a naming convention to generate the symbol name for the control block. This naming convention calls for the concatenation of:

- A prefix
- One or two fixed-length numeric values

Following are three examples of symbols that IPCS considers to be special symbols.

| Symbol | Associated Data and Example |
|---------------|--|
| ASCBnnnnn | The address space control block for address space nnnnn, STRUCTURE(ASCB). The nnnnn is decimal. (An example, ASCB1) |
| TCBnnnnnaaaaa | <p>The task control block for address space nnnnn, in position aaaaa in the priority queue, STRUCTURE(TCB). The nnnnn is decimal.</p> <p>The highest priority TCB in address space 1 is TCB00001AAAAA; the next TCB on the queue is TCB00001AAAAB,</p> <p>The last two characters in this name are alphabetic and range from AAAAA through AZZZZ, BAAAA, ... BZZZZ,</p> <p>(An example, TCB1A)</p> |
| UCBddd | The unit control block for device ddd, STRUCTURE(UCB). ddd designates the device address in hexadecimal. (An example, UCB1D0) |

When special symbols are entered as an argument to an IPCS subcommand, leading zero digits need not be entered. In base-26 notation, the letter “A” functions as the zero digit. IPCS recognizes the symbols without regard to whether leading zero digits are present. The symbols are converted to the standardized internal format for access to the symbol table.

Special symbols tend to be more legible to the human eye when leading zeros are removed, so IPCS strips leading zeros from the prefix prior to displaying any special symbol. When special symbols are entered containing leading “A”s, IPCS also removes them from the prefix before displaying the symbol.

See [z/OS MVS IPCS Commands](#) for a complete list of special symbols supported by IPCS.

Subcommands for Maintaining the Symbol Table

IPCS provides five subcommands to maintain the symbol table:

| When You Want to | Use this Subcommand |
|---|---------------------|
| Delete symbols from the symbol table | DROPSYM |
| Create a symbol with a user-defined name | EQUATE |
| List symbols and their attributes in the symbol table | LISTSYM |
| Renumber all stack symbols in the symbol table | RENUM |
| Add a pointer (symbol Znnnnn) to the IPCS pointer stack | STACK |

See [z/OS MVS IPCS Commands](#) for the syntax and examples of each subcommand.

Creating Your Own Symbol Definitions

Consider creating your own symbol definitions when a subcommand that examines dump data cannot locate a key block of storage. IPCS defines the appropriate symbols on demand and enters them in the symbol table for repetitive reference to the data. Occasionally, however, dumps are generated in which damage to low central storage or other factors prevent IPCS from successfully locating such data. If this occurs, you can define any symbol that is normally defined automatically by IPCS.

When a subcommand cannot locate a key block of storage, the subcommand ends and produces a message identifying the block that could not be accessed. In cases where you can identify the storage location of the required block, you can use the EQUATE subcommand to define its location for entry into the IPCS symbol table.

For example, if dump initialization fails and a message indicates that the segment table for the master address space is defective, use the EQUATE subcommand to define its symbol. Determine the segment table address using another dump that was taken on the same system at the same system level. If the segment table for the master address space is located at X'5D7C00' in central storage, with a length of 4 bytes and a dimension of 256 bytes, define the segment table by entering:

```
equate sgt001 5d7c00. absolute length(4) dim(256)
```

Here is another example. Assuming the communications vector table (CVT) is located in a dump at location X'F3C0' in absolute storage, you can define the CVT by entering:

```
equate cvt f3c0. absolute
```

Note: Processing for certain IPCS subcommands, such as ASMCHECK, COMCHECK, and IOSCHECK, will only accept explicit symbol definitions in virtual storage.

Validating Your Own Symbol Definitions

When IPCS has defined a symbol on demand, IPCS automatically validates the block of storage represented by that symbol before placing the symbol in the symbol table. You can specifically request validation of the data defined by a symbol by designating the data type in a STRUCTURE, MODULE, or AREA parameter.

In the following example, the STRUCTURE parameter designates the data type and requests validation:

```
equate CVT f3c0. absolute structure(cvt)
```

If IPCS detects serious defects during validation, IPCS does not define the symbol. In the example above, IPCS subcommands that require access to the CVT would issue a message that the symbol is needed and then end, rather than use damaged data.

You can use the LIST subcommand to examine the storage to determine the extent of damage to the block.

```
list f3c0. absolute length(1260)
```

If you determine that the damage will not affect the results of the IPCS subcommands to be run, you can force the definition into the symbol table by omitting the data type from the STRUCTURE parameter:

```
equate cvt f3c0. absolute structure
```

IPCS subcommands use the established definition without further validation, and the results of those subcommands are valid as long as the subcommands use fields that are not damaged. **If an IPCS subcommand uses damaged data, the subcommand might produce misleading diagnostic messages.**

Note: If a data area has an acronym and that acronym has been damaged, the CBFORMAT subcommand cannot format the damaged data area because the area fails the acronym check.

Listing Symbol Definitions

To display symbols and their addresses and attributes from the symbol table, use the LISTSYM subcommand. Parameters control the amount of symbols to be displayed.

To display some or all of the entries in the symbol table for a given dump, use the LISTSYM subcommand as follows:

```
listsym * print terminal
```

All the symbols associated with the current dump data set are displayed at your terminal and directed to the IPCSPRNT data set. The parameters PRINT and TERMINAL are not necessary unless they override the defaults. Following is an example of the output.

```

SYMBOL      ADDRESS  ATTRIBUTES
ASCB1       FCBA00. NOCPU ASID(X'0001') LENGTH(416) STRUCTURE(ASCB) NODROP
ASCB4       F2ED80. NOCPU ASID(X'0001') LENGTH(416) STRUCTURE(ASCB) NODROP
ASXB1       FCBC58. NOCPU ASID(X'0001') LENGTH(232) STRUCTURE(ASXB) NODROP
COMMON      B00000. CPU(X'00') ASID(X'0001') LENGTH(5242880) AREA(COMMON)
              NODROP
CVT          FCD4E8. NOCPU ASID(X'0001') POSITION(X'-0018') LENGTH(1288)
              STRUCTURE(CVT) NODROP

```

To list the address and attributes currently associated with the X symbol, enter:

```
listsym x
```

Deleting Symbol Definitions

To delete one or more symbols from the symbol table, use the DROPSYM subcommand. If the space allocated for your dump directory data set is full, delete some entries from your symbol table. Symbols with the NODROP attribute can be deleted with the DROPSYM subcommand only if you use the PURGE parameter. You do not need PURGE to delete a symbol with the DROP attribute.

Referring to the preceding LISTSYM output, delete the symbol ASCB4 by entering:

```
dropsym ascb4 purge
```

Notice that PURGE is specified because the symbol has the NODROP attribute.

If you delete X, IPCS may recreate it with an address of zero, using the default ASID and length attributes.

Adding and Deleting Symbols from the Pointer Stack

Within the symbol table is the IPCS pointer stack. The pointer stack consists of pointers with symbols in the form Znnnnn. IPCS determines the number of the symbol.

The pointer panel in the BROWSE option of the IPCS dialog uses the same pointer stack. Any changes made to the stack in line mode will change the pointers in the IPCS dialog.

To add a pointer to the IPCS pointer stack, use the STACK subcommand. For example, to add a pointer to address X'FFFF' in the dump, enter:

```
stack ffff. nodrop
```

You can view the pointers on the stack by using the pointer panel of the BROWSE option of the IPCS dialog or by using the LISTSYM subcommand. For example, to list all symbols from X to Z00010 inclusive, enter:

```
listsym (x:z00010)
```

IPCS displays the following list of symbols:

```

X          36000. CPU(X'00') ASID(X'0001') LENGTH(2147262464) AREA DROP
Z1          0. CPU(X'00') ASID(X'0001') LENGTH(4) AREA DROP
Z2         5555. CPU(X'00') ASID(X'0001') LENGTH(4) AREA DROP
Z3         FFFF. CPU(X'00') ASID(X'0001') LENGTH(4) AREA NODROP
4 DEFINITIONS LISTED

```

To delete a symbol from the pointer stack, use the DROPSYM subcommand, then use the RENUM subcommand to renumber all the stack symbols in the symbol table. Renumbering the symbols will return them to contiguous ascending order.

The following example, which was run after the previous LISTSYM and STACK subcommands, renumbers the pointer stack symbols and displays a message indicating the number of stack entries and the number of stack entries renumbered:

```
renum summary
```

```
The stack contains 3 entries, none of which were renumbered
```

Using the Storage Map

IPCS maintains a map of the dumped storage in the source description for a dump in the dump directory. IPCS creates the storage map as you enter subcommands. The storage map contains information that makes future references to that dump faster and more efficient.

The storage map is a list of dump addresses and data types. IPCS organizes the storage map for a dump by addresses in ascending order, then by address space, address within the address space, and data type. The data type describes the contents of the storage area at the address with which the data type is associated. The storage area can contain a control block, a module, or an area of storage such as an I/O buffer.

Each storage map entry contains a pointer to the dump records that contain the corresponding storage. An entry is added to the storage map whenever a name is specified with the STRUCTURE parameter (indicating a data area or control block), the MODULE parameter (indicating a module), or the AREA parameter (indicating an area in storage) on subcommands that accept these data description parameters.

Note: IPCS does not create a storage map when you process active storage (specified by the ACTIVE, MAIN, or STORAGE parameter).

Storage Map Entries Created by IPCS

During dump initialization, IPCS creates storage map entries in the storage map. Various IPCS subcommands also create entries when you specify an area with a data description parameter of STRUCTURE(name), MODULE(name), or AREA(name) or when the subcommand locates or validates such an area itself.

IPCS validates a control block by checking various attributes and characteristics to ensure that it is the control block that it is supposed to be. This means checking the control block's boundary alignment and the offset of certain special fields in it; such as acronyms, special values, masks, counts, and pointers to other control blocks, which may themselves be validated.

Be aware that IPCS does not validate every data area it finds. There are some data areas for which the subcommands make storage map entries but do not validate. For a list of the data areas supported, see the appendix in the [z/OS MVS IPCS Commands](#).

The following subcommand validates the data at X'51368' as a TCB:

```
equate failingtcb 51368. length(360) structure(tcb)
```

If it checks as a TCB, IPCS creates a storage map entry in the storage map for address X'51368'. The parameter STRUCTURE ensures that a storage map entry is created. IPCS also creates a symbol in the symbol table for FAILINGTCB.

The following subcommand explicitly verifies that the storage at X'522C0' is a TCB and makes a symbol table entry for MYTCB and a storage map entry for location X'522C0':

```
equate mytcb 522c0. structure(tcb)
```

In verifying the TCB, IPCS checks various pointers in the TCB to other control blocks, such as request blocks (RBs), contents directory entries (CDEs), etc. In the process, these control blocks may also be validated and entered in the storage map but not in the symbol table.

Subcommands for Maintaining the Storage Map

IPCS provides these subcommands to maintain and query the storage map:

| When You Want to | Use this Subcommand |
|---|---------------------|
| Delete storage map entries | DROPMAP |
| List storage map entries | LISTMAP |
| Validate control blocks | SCAN |
| Create storage map entries for address spaces satisfying certain selection criteria | SELECT subcommand |

See [z/OS MVS IPCS Commands](#) for syntax and examples of each subcommand.

Deleting Storage Map Entries

Delete storage map entries with the DROPMAP subcommand when the space allocated to your dump directory is depleted or the dump that the storage map entries represent is no longer associated with a problem on which you are working (the dump may still be associated with a problem on which another user is working).

The DROPMAP subcommand can delete some or all of the storage map entries for a dump. For example, to delete all storage map entries for the current dump that originate between the virtual addresses X'0' and X'50000' inclusive, enter:

```
dropmap range(0.:50000.)
```

Listing Storage Map Entries

List storage map entries with the LISTMAP subcommand. You can display some or all of the entries in the storage map. The LISTMAP subcommand lists the entries for a specified address space or range of addresses.

The following output lists the addresses and data types of some storage map entries after entering the following subcommand:

```
listmap
```

```
LIST 00000000. NOCPU ASID(X'0001') LENGTH(4096) MODULE(IEAVFX00)
LIST 00000000. NOCPU ASID(X'0001') LENGTH(4096) AREA(PRIVATE)
LIST 001455C8. NOCPU ASID(X'0001') LENGTH(4) STRUCTURE(RMCT)
LIST 00ADE970. NOCPU ASID(X'0001') LENGTH(4) STRUCTURE(DEB)
LIST 00AEDCF0. NOCPU ASID(X'0001') POSITION(X'-0020') LENGTH(408) STRUCTURE(TCB)
```

Validating Control Blocks

The process of validating one control block and following its pointers to other control blocks is called a scan probe. To do a scan probe, use the SCAN subcommand.

IPCS initiates SCAN subcommand processing from the storage map. SCAN validates control blocks that are listed in the storage map and that are within the specified address range. As it does this, SCAN makes new map entries for control blocks, modules, and areas pointed to by the block being validated. SCAN always has storage map entries to work with because dump initialization creates the original entries for a dump.

Suppose the map for a new dump contains only the entries for the CVT and the private area. To make one pass through the storage map, enter:

```
scan
```

If SCAN finds a structure that it maps, SCAN validates it. SCAN never validates a specific storage map entry a second time. Subsequent passes through the map only validate those entries not already completely validated.

If a large number for the DEPTH parameter were used in the above example, one scan probe could map all the areas and control blocks in a dump.

Creating Storage Map Entries for Address Spaces

Generate storage map entries that describe address spaces with the SELECT subcommand. These entries include the address of the address space, ASID, length, and AREA data type.

The following subcommand produces storage map entries that describe the selected address spaces:

```
select current error joblist(*master* jes3) nolist
```

Note: These entries can be accessed using the IPCS EVALMAP subcommand. The BLSCMAP CLIST in SYS1.SBLSCLI0 provides an example.

Changing Your Current User Dump Directory

While you can use only one user dump directory at a time, you can have several directories. You might want to arrange sources in groups and place each group in a different directory. For example, if you want to group all the data about a problem in several systems in a sysplex in one directory, you would copy the source descriptions for the dumps and traces into one directory. To use this single-problem directory, you would need to change from your current user dump directory to the single-problem directory.

The following procedure explains how to change the current dump directory for an IPCS dialog session. The examples in this procedure use the IPCS Subcommand Entry panel of the IPCS dialog.

1. Close all of the source data sets that are open for IPCS processing by entering the CLOSE subcommand as follows:

```
----- IPCS Subcommand Entry -----  
Enter a free-form IPCS subcommand, CLIST or REXX exec invocation below:  
  
===> close all
```

2. Close the current dump directory by entering the CLOSE subcommand as follows:

```
----- IPCS Subcommand Entry -----  
Enter a free-form IPCS subcommand, CLIST or REXX exec invocation below:  
  
===> close file(ipcsddir)
```

3. Use BLSCDDIR CLIST to allocate the new current dump directory. If the dump directory data set does not exist, IPCS will create a new VSAM data set.

For example, the following command asks IPCS to make IPCSU1.NEWDDIR the new current dump directory for the IPCS session. Both the DSNAME and VOLUME parameters override defaults in the BLSCDDIR CLIST:

```
----- IPCS Subcommand Entry -----  
Enter a free-form IPCS subcommand, CLIST or REXX exec invocation below:  
  
===> %blscddir dsname(ipcsu1.newddir) volume(vsam17)
```

If IPCSU1.NEWDDIR is an existing VSAM data set, IPCS uses the directory. If an IPCSU1.NEWDDIR VSAM data set does not exist, IPCS creates the new directory in a VSAM data set on volume VSAM17.

Administering the Sysplex Dump Directory

The sysplex dump directory can be administered by installation personnel who have READ and WRITE access using IPCS interactively or using batch TSO/E jobs.

An administrator can edit the sysplex dump directory. For example:

- Select from the source descriptions a dump or trace that needs to be screened by a person.
- Identify the person assigned to perform analysis for a source that requires specialized skills. For example, assign a source to John Doe with:

```
literal analyst c'John Doe'
```

- Copy a source description to a user dump directory or an archive data set that has a dump directory format.
- Delete a source description that is old or unneeded.

IPCS provides the following subcommands useful in managing the sysplex dump directory.

| When You Want to | Use this Subcommand |
|---|---------------------|
| Associate information with a dump. | LITERAL |
| Delete erroneous or obsolete information in the source description for the dump. | DROPDYM |
| Identify data sets containing unformatted dumps or traces. Only SVC dumps are automatically added to the sysplex dump directory. | ADDSDUMP |
| Obtain information from the sysplex dump directory | EVALDUMP, EVALSYSM |
| Copy a source description from the sysplex dump directory to the user dump directory of the person who will analyze the dump, or to an archive or history log | COPYDDIR |
| Delete a source description for a dump or trace no longer needed | DROPDUMP |

See [z/OS MVS IPCS Commands](#) for the syntax and examples of each subcommand.

Chapter 5. Describing Storage in a Dump

Certain IPCS subcommands used in dump analysis require a description of the data they are to process. Data description parameters tell a dump analysis subcommand the following:

- Where the data item is
- How big it is
- How to process the address
- What format to use when displaying or printing the data
- Whether the data is an array or a scalar

You can also associate a descriptive remark with the data item.

Such attributes are used by dump analysis subcommands to process the data and are also used by you to describe the data for displaying, printing, and your own future reference.

You would specify these attributes when you are displaying or printing data, creating a symbol table entry for it, or using it in an operation performed by a subcommand. The format attributes and the remark are usually used only when displaying or printing the data or when creating a symbol table entry for it. This latter case allows you to include the data item's attributes in the symbol table entry for future reference.

Instead of specifying data description parameters, you can use the X symbol, if it is appropriate. X is the symbol defined by IPCS to represent the current address in the current source. Various subcommands modify X during their processing. If the storage you want to process is addressed by X, you may use it instead of more complicated data description parameters.

This chapter contains:

- [“Categories of Data Description Parameters” on page 99](#)
- [“Specifying Data Description Parameters” on page 101](#)
- [“Resolving Default Data Description Parameters” on page 101](#)
- [“Using Indirect Addressing with Symbols” on page 102](#)

Categories of Data Description Parameters

Data description parameters fall into five categories:

- **Address, LENGTH, and POSITIONS parameters** describe the logical and physical location and the size of the data.
- **Address processing parameters** describe how the subcommand is to process the address. For example, the parameters indicate whether to simulate prefixing or dynamic address translation, and whether the dump source is in main storage or a data set.
- **Attribute parameters** describe the type of data and how it is to be displayed or printed.
- **Array parameters** describe the data as an array and define subscripts and dimensions or indicate that the data should not be treated as an array.
- **Remark parameters** associates remark text with data or indicates that there is no remark text.

The address and length parameters specify the location and size of the data item. The following are all valid addresses or address expressions:

ADDR1

is the symbolic address associated with the symbol ADDR1.

X

is the symbolic address associated with the current address in the dump.

+73

is the relative address X'73' bytes beyond the current address.

4C.

is a literal address at location X'4C'.

7R

is general-purpose register 7.

6D

is floating point register 6 with double precision.

6C.%

is a 24-bit indirect address, the location up to $2^{24}-1$ pointed from location X'6C'.

CVT+230?

is a 31-bit indirect address, the location up to $2^{31}-1$ addressed by the CVT field at offset +230 (field CVTGDA).

ADDR1-73n+4C

is an address expression, ADDR1 minus 73 (decimal) plus X'4C'.

Two typical uses of the full range of data display parameters are the EQUATE and LIST subcommands. For example, the following subcommand creates a symbol table entry for the symbol ABC:

```
equate abc a72f4. asid(1) length(80) area scalar +
  remark('input params from EXEC statement')
```

The symbol is associated with an 80-byte area beginning at X'A72F4'. ASID indicates that this address is virtual and IPCS simulates dynamic address translation for ASID(1). The AREA, SCALAR, and REMARK parameters specify information about this area that you want for future reference. AREA indicates that it is neither a module nor a control block. SCALAR indicates it is a single block of storage, not an array. The REMARK is your description of the 80-byte area.

Having created the symbol, you can now display its storage with the LIST subcommand. For example, the following subcommand displays the 80-byte buffer:

```
list abc
```

To display that storage without first defining the symbol, enter:

```
list a72f4. asid(1) length(x'50') area scalar
```

This subcommand displays the same data (except the remark), in the same format, as the previous example. Note that the length is defined in hexadecimal.

If you want to see the 20-byte field following the buffer and you want it displayed in character format, enter:

```
list abc+80n length(20) c
```

Another way to accomplish the same thing is to enter:

```
equate x abc+80n length(x'14')
list x c
```

This example sets X, the current address, to the end of the buffer and specifies a length of X'14'. The LIST subcommand then displays this area in character format.

Suppose you have located a segment table at X'7FFFD000'. If you want to display entries for segments six through ten, enter:

```
list 7FFFD018. length(4) entries(6:10)
```

This displays the five segment table entries beginning at X'5A2418' (each segment table entry is four bytes). The total length of the five entries is 20 bytes.

To list the contents of the general-purpose registers as they were at the time of the dump, enter:

```
list 0r:15r terminal
```

This displays all 16 general-purpose registers at your terminal.

To display the four floating-point double precision registers in hexadecimal, specify:

```
list 0d:6d
```

Note that when a symbol is assigned an offset, the offset is stored separately and is not added to the base address. Thus, the following subcommand creates a symbol record called A representing an address of 1000 and an offset of 10:

```
equate a 1000.+10
```

In a similar manner, the following subcommand uses only the **address** portion of A (1000) and creates a symbol, B, with an address of 1000 and an offset of 20:

```
equate b a+20
```

This design is used for the concept of fields within control blocks. Each field is considered at an offset from the beginning of the control block, not at an offset from another field.

Specifying Data Description Parameters

The syntax diagrams for IPCS subcommands in the *z/OS MVS IPCS Commands* specifies data description parameters as follows:

```
data-descr
```

This indicates that data description parameters are acceptable on a particular subcommand. Several subcommands place additional restrictions on the use of data description parameters. These restrictions are indicated in the list of parameter descriptions.

Resolving Default Data Description Parameters

An IPCS subcommand has several sources of defaults for data description parameters. A subcommand resolves the parameters by the following steps, in descending priority order.

1. The parameters specified on the subcommand.

If the default ASID is 2 and you specify ASID 7, the subcommand uses ASID 7.

2. The attributes of the symbol, if one is used.

Suppose the default ASID is 6 and you enter the following subcommands:

```
equate aaa 84. asid(2) length(30)
list aaa length(20)
```

LIST displays 20 bytes starting from X'84' in address space 2. Even though the default ASID is 6 for both subcommands, since you specified ASID 2 for the symbol AAA, when you omit the ASID parameter on the LIST subcommand, it uses the attribute of the symbol.

Consider the following example:

```
list aaa+4
```

Because location AAA+4 is within the area represented by AAA, you would get 26 bytes starting from location X'88' in address space 2.

Compare that to the result of entering:

```
list aaa+50
```

Because AAA+50 is outside the area represented by AAA, the number of bytes displayed is determined by the default length rather than the length associated with AAA. The data displayed begins at location X'D4'.

3. The defaults specified by the SETDEF subcommand.

If the default length is 50 and you create a symbol and specify no length, the EQUATE subcommand records a length of 50.

4. Defaults built into the IPCS subcommand, such as the AREA and SCALAR attributes.

Using Indirect Addressing with Symbols

When using symbols for indirect addressing, be sure to allow for symbols that have an offset. These symbols have an explicit +0 origin point. For example, to address an RB from symbol 'TCBnnna', use TCBnnnaa+0%. Not specifying the +0 results in the origin point being set at the beginning of the TCB prefix (displacement X'-20').

Chapter 6. Using IPCS REXX EXECs and CLISTs

IPCS subcommands can be entered from a REXX EXEC or CLIST. IPCS supplies a set of REXX EXECs and CLISTs in system library SYS1.SBLSCLI0. Your installation might also provide REXX EXECs and CLISTs. The ones that are provided by IPCS allow you to:

- Obtain a standard set of memory dump analysis reports.
- Print system storage areas
- Customize an IPCS session.
- Write your own IPCS REXX EXECs and CLISTs.

The system must have TSO/E to run IPCS REXX EXECs and CLISTs.



CAUTION:

IPCS programming environment under ISPF

Using the IPCS programming environment under ISPF is complicated. Two separate environments are involved: a command variable pool and a program variable pool. These affect how ISPF variables are available to any particular environment. Since IPCS starts in the program environment under ISPF (using SELECT PGM(BLSG)), the first time a command environment script runs (a REXX exec or TSO CLIST), it runs with (and updates) the program variable pool. When that command starts another one underneath it, that 'new' command refers to an uninitialized command variable pool (not the initialized program variable pool). So when a service obtains data from a variable pool (like a VGET), it searches the command variable pool and does not find any variables available, setting the rc=8. Subsequent commands can find a value if that second command-placed data into the 'new' command variable pool, but none of those commands find the data in the original program variable pool.

To get past this restriction in the ISPF variable pool processing domain, use a second exec, which creates a command variable pool by starting the SELECT CMD interface (for instance, ISPEXEC "SELECT CMD(PMTP2)") then PMTP2 performs the VPUT where the variable does get stored, for a subsequent VGET to reference the initialized variable in the command variable pool.

This chapter contains:

- [“Invoking REXX EXECs and CLISTs from an IPCS Session” on page 103](#)
- [“Starting an IPCS Session from a REXX EXEC or CLIST” on page 105](#)
- [“Using IPCS-Supplied CLISTs to Analyze Dumps” on page 105](#)
- [“Using IPCS-Supplied CLISTs to Print Dump Analysis Reports” on page 107](#)
- [“Using an IPCS-Supplied CLIST to Print Dump Storage” on page 107](#)
- [“IPCS-Supplied Sample REXX EXECs and CLISTs” on page 108](#)
- [“Writing IPCS REXX EXECs” on page 109](#)
- [“Techniques for Analyzing Dumps Using IPCS REXX EXECs” on page 115](#)
- [“Writing IPCS CLISTs” on page 119](#)
- [“Controlling Output from REXX EXECs and CLISTs” on page 120](#)

Invoking REXX EXECs and CLISTs from an IPCS Session

The method for invoking a REXX EXEC or CLIST from an IPCS session is the same as from a TSO/E session. You can use the implicit notation, extended implicit notation, or the EXEC command. IPCS recommends that you use the extended implicit notation (that is, precede the REXX EXEC or CLIST name with a percent sign) to invoke a REXX EXEC or CLIST from an IPCS session because it:

- Reduces the chances of IPCS confusing a REXX EXEC or CLIST name with an IPCS subcommand

- Results in better performance for the REXX EXEC or CLIST.

The following example shows one method for invoking a REXX EXEC or CLIST from an IPCS session.

Example: Invoking a REXX EXEC or CLIST

IPCS supplies a CLIST named BLSCEPTR in system library SYS1.SBLSCLI0. This CLIST runs the save area chain in a dump. To invoke BLSCEPTR from the IPCS dialog, do the following:

1. From the IPCS Primary Option Menu of the IPCS dialog, choose the COMMAND option. IPCS displays the Subcommand Entry panel.
2. On the command line, enter:

```
====> %BLSCEPTR
```

IPCS runs BLSCEPTR against the default dump data set. When the CLIST completes processing, IPCS displays a report on the dump display reporter panel.

You can invoke a REXX EXEC or CLIST from the command or option line of any other IPCS dialog panel. Preface the REXX EXEC or CLIST name with the IPCS primary command as follows:

```
----- IPCS PRIMARY OPTION MENU -----
OPTION ====>  IPCS %BLSCEPTR

      0  DEFAULTS      - Specify default dump and options      *****
      1  BROWSE        - Browse dump data set                  * USERID - IPCSU1
                                                              * DATE   - 95/05/14
```

You can invoke a REXX EXEC or CLIST from any IPCS session mode — batch, line, or dialog. IPCS batch mode is recommended for REXX EXECs and CLISTs that take a long time to complete processing.

Use the SUBMIT option of the IPCS dialog to create a batch job that invokes a REXX EXEC or CLIST. After you fill in the parameters on the panel, IPCS automatically codes the JCL and submits the job. For an explanation of the SUBMIT option, see [“Option 5 — SUBMIT” on page 40](#). If you want to code the JCL by hand, [Chapter 8, “Using IPCS in Batch Mode,” on page 125](#) explains the restrictions for running IPCS in batch mode and gives an example of the JCL needed to run a REXX EXEC or CLIST in batch mode.

Sometimes the types of commands used in a CLIST restrict where you can invoke the CLIST. See [Table 5 on page 119](#) for a summary of those restrictions or refer to the description of the REXX EXEC or CLIST in [z/OS MVS IPCS Commands](#).

Using ALTLIB

You can use the TSO/E ALTLIB command to define application-level libraries of REXX EXECs and CLISTs available for an IPCS dialog session. IPCS will first search any ALTLIB application libraries for a REXX EXEC or CLIST before searching other libraries.

For example, suppose you have modified copies of several IPCS REXX EXECs and CLISTs in the data set IPCSU1.SBLSCLI0. To place IPCSU1.SBLSCLI0 ahead of SYS1.SBLSCLI0 in the search order of REXX EXEC and CLIST libraries, choose the COMMAND option from the IPCS Primary Option Menu and enter the ALTLIB command as follows:

```
----- IPCS Subcommand Entry -----
Enter a free-form IPCS subcommand, CLIST or REXX EXEC invocation below:

====>  ALTLIB ACTIVATE APPLICATION(CLIST) DATASET('IPCSU1.SBLSCLI0')
```

From all other IPCS dialog panels, you must preface the ALTLIB command with the IPCS primary command as follows:

```

----- IPCS PRIMARY OPTION MENU -----
OPTION ==> IPCS ALTLIB ACTIVATE APPLICATION(CLIST) DA('IPCSU1.SBLSCLI0')
          0  DEFAULTS      - Specify default dump and options
          1  BROWSE        - Browse dump data set
                                *****
                                * USERID - IPCSU1
                                * DATE   - 95/05/14

```

An ALTLIB environment is unique for each IPCS dialog screen. If you split the screen and start another IPCS dialog session, the first ALTLIB environment does not apply to both screens. You can define another ALTLIB environment for the second screen.

When you exit the IPCS dialog, the ALTLIB environment for that IPCS session disappears. See the ALTLIB command in [z/OS MVS IPCS Commands](#) for more information.

Starting an IPCS Session from a REXX EXEC or CLIST

You can enter the TSO/E IPCS command from a REXX EXEC to start an IPCS session. The EXEC remains active during the IPCS session. The EXEC continues to the next instruction following the IPCS command only after you enter the END subcommand to end the IPCS session.

You cannot enter subcommands after the IPCS command in a REXX EXEC. The EXEC must queue the subcommands to the REXX data stack before entering the IPCS command. IPCS pulls any subcommands from the data stack after it starts the session. Therefore, do not address the IPCS host command environment when you enter the IPCS command from a REXX EXEC. [Figure 48 on page 105](#) shows how to start an IPCS session and enter subcommands from a REXX EXEC.

```

/* REXX */
Address TSO

/* Place IPCS subcommands on the stack */
Queue 'SETDEF' /* Display the IPCS session defaults */

/* Enter the IPCS command. When IPCS completes */
/* initialization, it pulls all subcommands from */
/* the stack. */
'IPCS NOPARM'

exit

```

Figure 48. Example of Starting an IPCS Session from a REXX EXEC

For most EXECs you should leave out the IPCS command and use the IPCS host command environment. If you must use the IPCS command in a procedure, CLISTs can check return codes from the IPCS command and each subcommand. A CLIST does not wait for the END subcommand to complete processing of the IPCS command. [Figure 49 on page 105](#) shows an example of starting an IPCS session from a CLIST.

```

IPCS NOPARM          /* Enter the IPCS command */
                    /* to start an IPCS session */
IF &LASTCC>8 THEN EXIT CODE(&MAXCC)

SETDEF              /* Display the IPCS session defaults */

```

Figure 49. Example of Starting an IPCS Session from a CLIST

See “[SIGNAL ON HALT and TRACE OFF Instructions](#)” on [page 110](#) for more information about checking IPCS subcommand return codes in a REXX EXEC.

Using IPCS-Supplied CLISTs to Analyze Dumps

When analyzing a dump, you start by entering subcommands to collect information about the dump. A REXX EXEC or CLIST can combine these subcommands to analyze a dump. IPCS provides several standard dump analysis CLISTs in system library SYS1.SBLSCLI0. Your installation may provide others.

CLISTs that Perform Initial Dump Analysis

IPCS provides a set of initial dump analysis CLISTs through the IPCS dialog. These initial dump analysis CLISTs create *screening* dump reports. A screening dump report usually provides enough information to identify any known problems for which service has not yet been applied. The report might be sufficient to identify problems for which maintenance is available but has not been installed.

The following table lists the initial dump analysis CLISTs:

| IPCS Dialog Options: | Invokes This CLIST: | To Process: |
|----------------------|---------------------|---|
| SUBMIT -> SADUMP | BLSCSCAN | Stand-alone dump stored on tape |
| SUBMIT -> SVCDUMP | BLSCBSVB | SVC dump |
| SUBMIT -> SYSMDUMP | BLSCBSYB | SYSMDUMP dump |
| SUBMIT -> SUPPLEMENT | BLSCSCAN | Stand-alone dump stored on DASD You must specify the name BLSCSCAN on the panel. |

The SUBMIT option of the IPCS dialog creates the batch job to run these CLISTs. You fill in the parameters on the panel and IPCS automatically codes the JCL and submits the job. The screening dump report goes to a SYSOUT data set. See [“Option 5 — SUBMIT” on page 40](#) for a description of the SUBMIT option and [z/OS MVS IPCS Commands](#) for details on each of these initial dump analysis CLISTs.

Example of Obtaining an SVC Dump Screening Report

To start analyzing an SVC dump from the IPCS primary option menu:

1. Select the SUBMIT option. IPCS displays the Dump Batch Job Option menu.

```
----- IPCS MVS DUMP BATCH JOB OPTION MENU -----
OPTION  ===>

  1  SADUMP    - Prepare stand alone dump for analysis
  2  SVCDUMP   - Prepare SVC dump for analysis
  3  SYSMDUMP  - Prepare SYSMDUMP for analysis
  4  SUPPLEMENT - Perform supplementary dump analysis

JOB STATEMENT INFORMATION:  (Verify before proceeding)

====>
====>
====>
====>
====>
====>

Enter END to terminate batch job processing.
```

2. Under JOB STATEMENT INFORMATION, type the JCL for the JOB statement. This panel saves the information for the next time you submit a job.
3. Select the SVCDUMP option to analyze an SVC dump. IPCS displays a panel for you to specify job parameters.

```
----- Prepare SVC Dump for IPCS Analysis -----
COMMAND  ===>

Enter/verify parameters for the job.
Use ENTER to submit the job, END to terminate without job submission.

DATA SET NAME  ===>
DUMP DIRECTORY ===>
SYSOUT CLASS   ===>
```

4. Specify the name of the dump data set, the name of the dump directory, and the SYSOUT class, and press enter.

IPCS codes the rest of the JCL and submits the job. Remember that you cannot use the dump directory specified for the batch job until the job is complete.

The output from this job contains a series of reports from subcommands run by the BLSCSVCB CLIST. The VERBEXIT SYMPTOM subcommand, which runs at the end of the CLIST, formats any RETAIN symptoms that were generated by previous subcommands. The output should provide enough information for you to continue analyzing the dump through the IPCS dialog.

CLIST that Runs the Save Area Chain

The BLSCEPTR runs the forward chain of save areas, beginning with the error TCB. See [z/OS MVS IPCS Commands](#) for more information.

Using IPCS-Supplied CLISTs to Print Dump Analysis Reports

IPCS provides a set of CLISTs to print dump analysis reports. The CLISTs in the following sections must be invoked from TSO/E. (See “Using IPCS-Supplied CLISTs to Analyze Dumps” on page 105 to submit these reports from an IPCS session.) They are designed to be submitted as batch jobs using the cataloged procedure BLSJIPCS, alias IPCS. You can print these reports from SYSOUT or send the reports directly to IPCSPRNT data set. For more information, see [Chapter 8, “Using IPCS in Batch Mode,”](#) on page 125.

CLISTs that Print Screening Dump Reports

These CLISTs produce the same screening dump reports as do the initial dump analysis CLISTs invoked from the IPCS dialog. See [z/OS MVS IPCS Commands](#) for more information.

The following table lists the screening dump report CLISTs:

| Dump Type | Screening CLIST |
|------------------|-----------------|
| Stand-alone dump | BLSCBSAA |
| SVC dump | BLSCBSVA |
| SYSMDUMP dump | BLSCBSYA |

CLISTs that Print Detailed Dump Reports

These printing CLISTs format the same information as screening CLISTs, but they also format storage in the dump that is most often used for more detailed problem analysis. See [z/OS MVS IPCS Commands](#) for more information.

The following table lists the detailed dump report CLISTs:

| Dump Type | Printing CLIST |
|------------------|----------------|
| Stand-alone dump | BLSCBSAP |
| SVC dump | BLSCBSVP |
| SYSMDUMP dump | BLSCBSYP |

Note: BLSCBSAP copies a stand-alone dump tape to a cataloged dump data set. You must specify the tape on the IEFRDER DD statement in the cataloged procedure.

Using an IPCS-Supplied CLIST to Print Dump Storage

The BLSCPRNT CLIST prints unformatted storage from a dump. You can specify which storage areas it should print with parameters listed in [z/OS MVS IPCS Commands](#).

IPCS-Supplied Sample REXX EXECs and CLISTs

You can use a combination of IPCS subcommands, TSO/E commands, ISPF commands, and CLIST commands to create your own analysis and formatting routines. IPCS provides sample REXX execs and CLISTs to show how to use these commands and subcommands. The following sections describe these REXX EXECs and CLISTs.

Retrieving Data from a Dump

IPCS provides a series of subcommands to retrieve information for use in a REXX EXEC or CLIST. The following table lists those subcommands and the corresponding CLISTs in SYS1.SBLSCLI0 that give examples of how to use the subcommands.

| Subcommand | Purpose | Sample CLIST |
|------------|--|--------------|
| EVALDEF | Retrieve the SETDEF-defined defaults. | BLSCSETD |
| EVALDUMP | Retrieve a dump's description from the dump directory. | BLSCEDUM |
| EVALMAP | Retrieve an entry from the IPCS storage map. | BLSCEMAP |
| EVALSYM | Retrieve an entry from the IPCS symbol table. | BLSCESYM |

See [z/OS MVS IPCS Commands](#) for more information about each subcommand.

Processing Control Blocks

IPCS subcommands add information about control blocks to the symbol table and storage map. IPCS does this automatically as you process control blocks in a dump. To add your own control blocks to the symbol table and storage map, you can create a REXX EXEC or CLIST.

For example, you can write a CLIST that uses the RUNCHAIN subcommand to run an application's control block chains and the EQUATE subcommand to define those application control blocks to IPCS (i.e. - create entries in the symbol table).

The following table lists some subcommands used to process control blocks and dump data and the sample REXX EXEC and CLISTs in SYS1.SBLSCLI0 that show how to code these subcommands:

| Subcommand | Purpose | Sample REXX EXEC or CLIST |
|------------|---------------------------------------|---------------------------|
| COMPARE | Compare two pieces of data in a dump. | BLSCCOMP |
| RUNCHAIN | Run a chain of control blocks. | BLSCRNCH BLSCRNC2 |
| WHERE | Find the location of a load module. | BLSXWHER |

See [z/OS MVS IPCS Commands](#) for more information about each subcommand.

Printing Dump Storage

Usually you will use IPCS subcommands in REXX EXECs and CLISTs to format and analyze dump storage. However, you can add unformatted sections of dump storage to your dump report. IPCS provides a set of CLISTs in SYS1.SBLSCLI0 that can be invoked from your CLIST to print sections of storage. The following table summarizes these CLISTs:

| CLIST | System storage printed |
|----------|--|
| BLSCPCSA | The common storage area (CSA) and extended CSA (ESCA). |

| CLIST | System storage printed |
|----------|---|
| BLSCPNUC | The nucleus, extended nucleus, read-only nucleus, and dynamic-address-translation-off (DATOFF) nucleus. |
| BLSCPSQA | The global system queue area (SQA) and extended SQA (ESQA). |
| BLSCPRIV | The private and extended private areas. |

For an example of how to invoke these CLISTs from a CLIST, see the BLSCPRT CLIST in SYS1.SBLSCLI0.

Writing IPCS REXX EXECs

This section describes some techniques you might want to use for writing and debugging IPCS REXX EXECs.

Using IPCS and REXX Facilities for Writing REXX EXECs

Several facilities are available for use in writing a REXX EXEC. These facilities include both REXX instructions and IPCS subcommands.

The REXX instructions are:

- ADDRESS IPCS
Use ADDRESS IPCS to change the host command environment to IPCS.
- SIGNAL ON HALT and TRACE OFF
Use SIGNAL ON HALT and TRACE OFF to suppress unwanted error messages and trace records.

The IPCS subcommands are:

- EVALUATE
Use the EVALUATE subcommand to retrieve data from a dump and store the data in a REXX variable.
- NOTE
Use the NOTE subcommand for output from the REXX EXEC.

The following sections provide detailed descriptions of each of these facilities along with a summary, “Putting the Pieces Together” on page 114.

ADDRESS IPCS Instruction

To enter subcommands from a REXX EXEC, use the ADDRESS IPCS instruction. ADDRESS IPCS changes the host command environment to IPCS. The IPCS host command environment is available only when you run the EXEC from an IPCS session.

The example in [Figure 50 on page 109](#) is a simple REXX EXEC that changes the host command environment to IPCS for all commands and then enters the STATUS and LIST subcommands.

```
/* REXX */
Address IPCS
'STATUS'
if rc > 0 then signal Exit
'LIST 0. LENGTH(100)'
Exit: exit
```

Figure 50. Changing the Host Command Environment to IPCS for All Commands

You can also use ADDRESS IPCS for a single command. [Figure 51 on page 110](#) gives an example.

```

/* REXX */

Address IPCS 'STATUS'
if rc > 0 then signal Exit
Address IPCS 'LIST 0. LENGTH(100)'

Exit: exit

```

Figure 51. Changing the Host Command Environment to IPCS for a Single Command

Modes of IPCS Operation

REXX EXECs under TSO/E function in a single operating mode as is illustrated by Figure 50 on page 109. The EXEC shown in that figure is designed to function in READY mode and will not function correctly in any other mode. The IPCS command has the following modes:

- Parmlib members are processed during session initialization. In OS/390 Version 2 Release 10, IPCS introduces ADDRESS IPCS support for REXX EXECs in this mode.
- IPCS mode is the normal mode of operation throughout most of an IPCS session. All supported MVS/SP and z/OS releases accept ADDRESS IPCS requests in this mode.
- Trap stops establish their own mode of operation. All supported MVS/SP and z/OS releases accept ADDRESS IPCS requests in this mode.
- TSO/E mode allows most TSO/E commands and command procedures that can be used in READY mode to be run under IPCS. No ADDRESS IPCS support is intended for this mode.

Additional information

See [z/OS TSO/E REXX User's Guide](#) for more information about host command environments.

SIGNAL ON HALT and TRACE OFF Instructions

If the user exits an IPCS report before all subcommand processing completes, the system might produce REXX error messages or a REXX trace record. You can use the SIGNAL ON HALT instruction to suppress unwanted error messages, and you can use the TRACE OFF instruction to suppress a REXX trace record.

Your REXX EXECs should check the return code from each IPCS subcommand invocation. If the return code indicates an error, exit the EXEC or perform some other appropriate processing. The reasons for doing this are as follows:

- It stops the REXX EXEC from using potentially incorrect data retrieved by a subcommand.
- It allows an IPCS user to end an EXEC that produces output to a dump display reporter panel in the IPCS dialog. The user can end the EXEC before all subcommands have completed processing.

The example in Figure 52 on page 110 is a simple REXX EXEC that suppresses error messages and the REXX trace record, and checks the return code from the STATUS subcommand. If the user presses PF3 while the EXEC is running, the system returns to the panel from which the user issued the EXEC.

```

/* REXX */

Signal on halt

Trace off          /* Suppress display of unsuccessful
                    storage fetches once PF3 has been hit */

Address IPCS

'STATUS'
if rc > 0 then signal Exit
'LIST 0. LENGTH(4000)'
Signal Exit

Halt:

Exit: exit

```

Figure 52. Using SIGNAL ON HALT and TRACE OFF and Checking Return Code

See *z/OS MVS IPCS Commands* for subcommand return codes.

EVALUATE Subcommand

To retrieve data from a dump and store the data in a REXX variable, you can use the EVALUATE subcommand, along with an internal REXX function shown in [Figure 53 on page 112](#). This function, Obtain_Data, requires three arguments:

1. The address of the data in the dump
2. The offset from the address
3. The amount of data to retrieve

Obtain_Data returns to the caller a continuous block of dump data in a REXX variable.

Obtain_Data determines the most efficient use of the EVALUATE subcommand to retrieve the dump data. First it checks if the caller requests data from the same hexadecimal address that it has processed before. If so, it retrieves the data from the REXX stem variable *storage*. If the data is not available in *storage*, the function uses EVALUATE to retrieve 512 bytes of data (EVALUATE's limit) and stores the data in *storage*. It uses another variable, *buffer*, to build the continuous block of data that is returned to the caller. Obtain_Data handles requests greater than 512 bytes by concatenating each 512-byte block to *buffer*.

Note: Callers of Obtain_Data must request storage from the current SETDEF-defined address space default (ASID or CPU). If you change the address space default across invocations of Obtain_Data in an EXEC, reinitialize *storage* to ensure Obtain_Data returns the correct data.

```

Obtain_Data: procedure expose storage.
/*-----*/
/* Function: Obtain_Data */
/*
/* Retrieve data from the dump. Invoke the IPCS
/* EVALUATE subcommand as necessary to access
/* 512-byte blocks of data from the IPCS dump
/* source and store the data in variable
/* "storage." Callers of Obtain_Data must
/* request storage from the same address space.
/*
/* Input: Description of data to access:
/*
/* Hex address of data.
/* Decimal position from the hex address of the
/* first byte to access.
/* Decimal length of the data to access.
/*
/* Output: Requested data is returned.
/*-----*/

arg hex_address , dec_position , dec_length
trace off /* Suppress display of unsuccessful storage
fetches once PF3 has been hit */

Numeric digits(10)
ipcs_eval_limit = 512 /* The maximum number of bytes that
the IPCS EVALUATE subcommand can
access per invocation */
page_size = 4096 /* The size of a storage page */
first_index = dec_position % ipcs_eval_limit /* Determine the
first 512 byte increment */
last_index = (dec_position+dec_length) % ipcs_eval_limit /*
Determine the last 512 byte
increment */
buffer = '
do i= first_index to last_index /* For each 512 increment */
if storage.hex_address.i = '' then do

/*-----*/
/* If the data has not yet been accessed, access it. */
/*-----*/

hex_address_dot = ,
hex_address||. /* Indicate for IPCS that it is an
address */
"EVALUATE" hex_address_dot ,
"POSITION("i*ipcs_eval_limit")" ,
"LENGTH("ipcs_eval_limit")" ,
"REXX(STORAGE(X))" /* Access the data by invoking
the IPCS EVALUATE subcommand */

if rc > 0 then do

if rc > 0 then do
/*-----*/
/* If 512 bytes of data could not be accessed, determine*/
/* if the data spans across a page. If it does, attempt*/
/* to access the data that resides in the first page. */
/*-----*/

address = x2d(hex_address)+i*ipcs_eval_limit
new_length=page_size-(address//page_size)
if new_length=0 | new_length >= ipcs_eval_limit then
signal Access_Error
"EVALUATE" hex_address_dot ,
"POSITION("i*ipcs_eval_limit")" ,
"LENGTH("new_length")" ,
"REXX(STORAGE(X))" /* Access the data by invoking
the IPCS EVALUATE subcommand */
if rc > 0 then signal Access_Error
end
storage.hex_address.i = x /* Save the data in a
variable so that it only needs to
be accessed once */
end
buffer = buffer||storage.hex_address.i /* Augment the buffer
with the current data */
end /* For each 512 increment */
return_offset = (dec_position-first_index*ipcs_eval_limit)*2+1
return_length = dec_length*2
if return_offset-1 + return_length > length(buffer) then

/*-----*/
/* Do not attempt to return more than what is in the buffer. */
/*-----*/

signal Access_Error
return substr(buffer,return_offset,return_length) /* Return
the appropriate data from the
buffer */

```

Figure 53. Example of a REXX EXEC Function to Retrieve Dump Data

Invoking Obtain_Data

To make Obtain_Data work correctly in a REXX EXEC, the EXEC must do the following:

1. Clear the REXX variable *storage* at the beginning of the EXEC:

```
storage. = '
```

Obtain_Data uses this variable to store dump data.

2. Include a function similar to Put (see [Figure 54 on page 113](#)) in the exec.
3. Add the following code at the label Access_Error:

```
Access_Error: nop
GEN$='IPCS Evaluate subcommand unable to access storage'
Call Put                               /* Display GEN$          */
Exit 16
```

This code together with the Put function (see [Figure 54 on page 113](#)) handles errors encountered by Obtain_Data.

Example – Invoking Obtain_Data

Suppose you want to find the address of the link list table (LLT) in a dump. This can be done in two steps using Obtain_Data and information found in the z/OS MVS Data Areas information.

1. Find the address of the communications vector table (CVT). The prefix storage area (PSA), which is located at address zero of a dump, has a pointer to the CVT at offset X'10'. To retrieve a pointer, ask for four bytes of data:

```
cvt_address = obtain_data(0,x2d('10'),4)
```

2. At offset X'4DC' of the CVT is the address of the LLT:

```
lnklst_address = obtain_data(cvt_address,x2d('4DC'),4)
```

With the address of the LLT, you can continue to invoke Obtain_Data to retrieve the names of the data sets in the link list concatenation.

For best performance, REXX EXECs that retrieve dump data should make Obtain_Data (or a similar function) an internal function of each EXEC. Making Obtain_Data an external function would slow performance of the REXX EXEC.

NOTE Subcommand

Use the NOTE subcommand to display the results of your REXX EXEC. The NOTE subcommand allows you to control the output using the PRINT and TERMINAL parameters. As with any IPCS subcommand, your EXEC should check the return codes from each NOTE subcommand (see [“SIGNAL ON HALT and TRACE OFF Instructions” on page 110](#)).

If your REXX EXEC uses NOTE more than once to display output, consider creating an internal function to invoke NOTE. [Figure 54 on page 113](#) is an example of such a function.

```
Put: procedure expose GEN$
/*-----*/
/* Function: Put                                     */
/*-----*/
/*          Invoke the IPCS NOTE subcommand to transmit */
/*          data to the terminal, IPCS print data set or */
/*          both depending on the IPCS message routing */
/*          default.                                     */
/*-----*/
/* Input:    Data to transmit.                         */
/*-----*/
/* Output:   Data is transmitted.                      */
/*-----*/

"NOTE 'GEN$' ASIS"
if rc>0 then Signal Put_Error
return
```

Figure 54. Example of a REXX Function to Display Output

To handle any error from NOTE processing, a REXX EXEC needs the following lines in addition to the Put function:

```
Put_Error: nop
Retc = 16
Signal Exit

Exit: exit Retc
```

To invoke Put, the EXEC should set the variable GEN\$ equal to the output it wants to display. For example:

```
GEN$ = 'The dump data set name is 'data_set_name
Call Put
```

For more information about displaying IPCS output, see [“Controlling Output from REXX EXECs and CLISTS” on page 120.](#)

Putting the Pieces Together

The preceding sections give a few building blocks for creating IPCS REXX EXECs. To see a complete EXEC, look at the sample IPCS REXX EXEC BLSXWHER in system library SYS1.SBLSCLI0. BLSXWHER analyzes load modules in the private area of storage. Use BLSXWHER as a guide for writing your own IPCS REXX EXECs. Also, reference the information in [“Techniques for Analyzing Dumps Using IPCS REXX EXECs” on page 115](#) for more sample IPCS REXX dump analysis functions.

IPCS REXX EXECs can do more than analyze dumps. You can write execs to perform utility tasks in IPCS. Figure 55 on page 114 gives a simple EXEC that transfers information from a batch dump directory to a user's current dump directory.

```
/* REXX */

/*-----*/
/* Function: Copy a dump description from our installation's*/
/*           batch dump directory 'SYS1.DDIR' into the      */
/*           current dump directory and display a summary of*/
/*           the storage from the dump description.         */
/*-----*/
ARG dump_name

Address IPCS

"COPYDDIR INDSNAME('SYS1.DDIR') DSNAME("dump_name")"

exit rc
```

Figure 55. Sample IPCS REXX EXEC to Copy Dump Directory Information

Debugging IPCS REXX EXECs

IPCS provides attention processing for its host command environment. The attention processing makes available all of the REXX debug aids from an interactive IPCS session. During processing of an IPCS REXX EXEC, you can press the attention interrupt key to use the debug aids in attention mode.

The attention processing handles IPCS subcommands differently depending on the mode.

- In IPCS line mode, subcommands run to completion before an immediate command takes effect.
- In the IPCS dialog, the HI immediate command ends both the subcommand and the EXEC. For all other immediate commands, the IPCS subcommands run to completion before the immediate command takes effect.

For example, suppose you press the interrupt key while a REXX EXEC is running the SUMMARY subcommand. The system reacts as follows:

- If the EXEC is run in IPCS line mode, the system displays message IRX0920I:

```
ENTER HI TO END, A NULL LINE TO CONTINUE, OR AN IMMEDIATE COMMAND
```

You respond by entering the HI immediate command. Because the SUMMARY subcommand has not completed processing, the system completes processing of the subcommand before ending the EXEC.

- If the EXEC is run in the IPCS dialog, the system displays the following message:

```
Enter HI to end, a null line, TIME, or an immediate command
```

You respond by entering the HI immediate command. The system ends the subcommand and the EXEC.

See [z/OS MVS IPCS Commands](#) for information about attention processing in IPCS and [z/OS TSO/E REXX Reference](#) for descriptions of the REXX debug aids.

Techniques for Analyzing Dumps Using IPCS REXX EXECs

This section gives examples of REXX functions that perform common dump analysis and formatting functions, such as how to:

- Add hexadecimal dump addresses
- Follow a pointer chain
- Format a control block
- Translate hexadecimal dump characters for display.

All examples in this section assume that the REXX EXEC has changed the host command environment to IPCS.

Adding Hexadecimal Dump Addresses

REXX provides several built-in hexadecimal conversion functions that are useful for IPCS REXX EXECs. For example, [Figure 56 on page 115](#) gives a simple function for adding two hexadecimal dump addresses:

```
HexAdd: Procedure
  Numeric digits 10
  Return D2X(X2D(Arg(1)) + X2D(Arg(2)))
```

Figure 56. A Function to Add Hexadecimal Dump Addresses

Note: When translated to decimal, the hexadecimal addresses can be larger than the REXX default of nine decimal digits. An EXEC that adds dump addresses should use the NUMERIC DIGITS instruction to change the default precision to 10 decimal digits.

Following a Pointer Chain

[Figure 57 on page 116](#) gives a recursive function for following a chain of task control blocks (TCB). The function, CheckTCB, invokes Obtain_Data (see [Figure 53 on page 112](#)) to retrieve the TCB pointers from the dump and Put (see [Figure 54 on page 113](#)) to display the addresses.

```

CheckTCB: Procedure expose storage.
TCBADDR= ARG(1)
GEN$ = " "
Call Put
Gen$ = "TCB at address "TCBADDR
Call Put
TCBNTC   = Obtain_Data(TCBADDR,128,4)
TCBLTC   = Obtain_Data(TCBADDR,136,4)
TCBRTWA  = Obtain_Data(TCBADDR,224,4)
TCBCMP   = Obtain_Data(TCBADDR,16,4)
Gen$ = " TCBntc...."TCBNTC
Call Put
Gen$ = " TCBltc...."TCBLTC
Call Put
Gen$ = " TCBrtwa..."TCBRTWA
Call Put
Gen$ = " TCBCmp...."TCBCMP
Call Put
/*****
If TCBLTC ^= "00000000"
Then
    Call CheckTCB TCBLTC
If TCBNTC ^= "00000000"
Then
    Call CheckTCB TCBNTC
Return

```

Figure 57. Example of a REXX EXEC Function to Follow a TCB Structure

If X'00F78400' is the address of a TCB, you would invoke this function as follows:

```
CheckTCB 00F78400
```

This simple function can be coded more efficiently using the RUNCHAIN subcommand to follow a chain of control blocks. A REXX function, however, allows you to customize the analysis of the control block chain. See [“Customizing Control Block Analysis” on page 117](#) for an example of customizing the CheckTCB function.

Formatting a Control Block

Figure 58 on page 117 gives a function that formats selected fields in the recovery termination manager 2 work area (RTM2WA). The function, ShowRTM2WA, invokes Obtain_Data (see Figure 53 on page 112) to retrieve the RTM2WA fields from the dump and Put (see Figure 54 on page 113) to display the fields.

```
ShowRTM2WA: procedure expose RTM2WAAddr storage.
```

```
RTM2CODE = Obtain_Data(RTM2WAAddr,28,4)
RTM2ER0  = Obtain_Data(RTM2WAAddr,60,4)
RTM2ER1  = Obtain_Data(RTM2WAAddr,64,4)
RTM2ER2  = Obtain_Data(RTM2WAAddr,68,4)
RTM2ER3  = Obtain_Data(RTM2WAAddr,72,4)
RTM2ER4  = Obtain_Data(RTM2WAAddr,76,4)
RTM2ER5  = Obtain_Data(RTM2WAAddr,80,4)
RTM2ER6  = Obtain_Data(RTM2WAAddr,84,4)
RTM2ER7  = Obtain_Data(RTM2WAAddr,88,4)
RTM2ER8  = Obtain_Data(RTM2WAAddr,92,4)
RTM2ER9  = Obtain_Data(RTM2WAAddr,96,4)
RTM2ER10 = Obtain_Data(RTM2WAAddr,100,4)
RTM2ER11 = Obtain_Data(RTM2WAAddr,104,4)
RTM2ER12 = Obtain_Data(RTM2WAAddr,108,4)
RTM2ER13 = Obtain_Data(RTM2WAAddr,112,4)
RTM2ER14 = Obtain_Data(RTM2WAAddr,116,4)
RTM2ER15 = Obtain_Data(RTM2WAAddr,120,4)
RTM2EPSW1 = Obtain_Data(RTM2WAAddr,124,4)
RTM2EPSW2 = Obtain_Data(RTM2WAAddr,128,4)
RTM2ILC1  = Obtain_Data(RTM2WAAddr,133,1)
RTM2INC1  = Obtain_Data(RTM2WAAddr,134,2)
RTM2TRAN  = Obtain_Data(RTM2WAAddr,136,4)
RTM2ABNM  = Obtain_Data(RTM2WAAddr,140,4)
GEN$ = " "
Call Put
GEN$ = " RTM2WA:" RTM2WAAddr " "
Call Put
GEN$ = " PSW....."RTM2EPSW1" "RTM2EPSW2
Call Put
GEN$ = " ILC....."RTM2ILC1
Call Put
GEN$ = " IntCd...."RTM2INC1
Call Put
GEN$ = " Code....."RTM2CODE
Call Put
GEN$ = " LoadMod.."RTM2ABNM
Call Put
GEN$ = " GPR0-3..."RTM2ER0 RTM2ER1 RTM2ER2 RTM2ER3
Call Put
GEN$ = " GPR4-7..."RTM2ER4 RTM2ER5 RTM2ER6 RTM2ER7
Call Put
GEN$ = " GPR8-11..."RTM2ER8 RTM2ER9 RTM2ER10 RTM2ER11
Call Put
GEN$ = " GPR12-5..."RTM2ER12 RTM2ER13 RTM2ER14 RTM2ER15
Call Put
Return
```

Figure 58. Example of a REXX EXEC Function to Format the RTM2WA

Customizing Control Block Analysis

The two preceding sections show how to code REXX functions to run a control block chain and format a control block. But IPCS already provides similar function through the CBFORMAT and RUNCHAIN subcommands. This section shows how to add to the CheckTCB function to customize the control block analysis beyond the ability of the CBFORMAT and RUNCHAIN subcommands.

Customizing the TCB Pointer Chain Function

When CheckTCB runs the TCB pointer chain (Figure 57 on page 116), the function can check each TCB for an associated RTM2WA. If an RTM2WA does exist, CheckTCB can call ShowRTM2WA (Figure 58 on page 117) to format it.

Figure 59 on page 118 gives the lines of REXX code needed to analyze the TCB completion codes and RTM2WA addresses for each TCB in the chain. Insert these lines after the line of asterisks in CheckTCB:

```

If TCBCMP ^= "00000000" & TCBRTWA = "00000000" Then
  Do
    GEN$ = " "
    Call Put
    GEN$ = "TCB at "TCBAddr" has non-zero completion code "TCBCMP","
    GEN$ = GEN$||" but TCBRTWA is zero."
    Call Put
    GEN$ = "A prior abend has probably"
    GEN$ = GEN$||" been handled by an ESTAE routine."
    Call Put
    GEN$ = " "
    Call Put
  End
If TCBCMP = "00000000" & TCBRTWA ^= "00000000" Then
  Do
    GEN$ = " "
    Call Put
    GEN$ = "TCB at "TCBAddr" has zero completion code,"
    GEN$ = GEN$||" but does have an RTM2WA."
    Call Put
    GEN$ = "This is most unusual."
    Call Put
    Call ShowRTM2WA TCBRTWA
  End
If TCBCMP ^= "00000000" & TCBRTWA ^= "00000000" Then
  Do
    GEN$ = " "
    Call Put
    GEN$ = "TCB at "TCBAddr" has non-zero completion code "TCBCMP"."
    Call Put
    Call ShowRTM2WA TCBRTWA
  End
End

```

Figure 59. Example of REXX Code to Analyze the Contents of a TCB

Translating Dump Characters

A dump contains a wide variety of hexadecimal characters. For example, the character X'05' is a horizontal tab character for certain display terminals. If the REXX EXEC tries to display this data, the X'05' appears as multiple blank positions on those displays.

Figure 60 on page 118 gives one method for handling unprintable dump characters by converting them to periods for display.

```

Readable: Procedure
  /* 0123456789ABCDEF */
  From = ""
  Do I = 0 To 255
    From = From||X2C(D2X(I))
  End
  To = " ..... "
  To = To" ..... "
  To = To" ..... "
  To = To" ..... "
  To = To" .....<(+| "
  To = To"&;.....!$*);^"
  To = To"-.....|,%>?"
  To = To".....:#@T=" " "
  To = To".abcdefghi....."
  To = To".jklmnopqr....."
  To = To"..stuvwxyz....."
  To = To"....."
  To = To".ABCDEFGHI....."
  To = To".JKLMNOPQR....."
  To = To"/.STUVWXYZ....."
  To = To"0123456789....."
  Return Translate(Arg(1),To,From)

```

Figure 60. A REXX EXEC Function for Translating Unprintable Dump Characters

Writing IPCS CLISTs

When writing CLISTs that contain TSO/E commands, you must consider the environment in which the CLIST will run when designing the CLIST. This is especially important when designing CLISTs that contain both TSO/E commands and IPCS subcommands.

When you invoke a CLIST from the IPCS dialog, the CLIST can use IPCS subcommands to place information in ISPF function pool dialog variables and can use the ISPEXEC subcommand to request dialog services that use the dialog variables. If TSO/E Release 2 is not installed, ISPF has no knowledge of any CLIST variables that are used in a CLIST initiated through this panel. Therefore, such CLISTs cannot reference ISPF variables.

Given this flexibility, the functions that can be used in a CLIST are dependent on the environment from which you invoke the CLIST. Before you write the CLIST, decide what environment you will typically invoke the CLIST from. Then review [Table 5 on page 119](#) for restrictions.

| Table 5. Considerations for designing and invoking IPCS CLISTs | |
|--|--|
| CLIST Contains: | Invoke as Follows: |
| IPCS subcommands | In IPCS line mode, batch mode, or through the IPCS dialog |
| TSO/E commands | <p>In IPCS line mode, batch mode, or through the IPCS dialog <i>with or without</i> the TSO subcommand:</p> <ol style="list-style-type: none">1. Invoke the CLIST <i>with</i> the TSO subcommand if either of the following are true:<ul style="list-style-type: none">• The CLIST contains a combination of TSO/E authorized commands and functions such as SYSOUTTRAP, and you are invoking the CLIST while in the IPCS dialog• One or more TSO/E commands in the CLIST have the same names as IPCS subcommands2. Invoke the CLIST <i>without</i> the TSO subcommand if the items listed in step “1” on page 119 are not true. |
| A combination of IPCS subcommands and TSO/E commands | <p>From IPCS line or batch mode, or through the IPCS dialog, there are three possible ways of invoking such a CLIST:</p> <ol style="list-style-type: none">1. Use the TSO subcommand. If invoked in this way, the IPCS subcommands in the CLIST must be invoked by using the BLSGSCMD dialog program.2. Invoke the CLIST <i>without</i> using the TSO subcommand. By invoking the CLIST in this way, you do not have to use the BLSGSCMD dialog program to invoke the IPCS subcommands within the CLIST. You can only invoke the CLIST in this way if the CLIST does not contain either of the following:<ul style="list-style-type: none">• TSO/E commands that have the same names as IPCS subcommands.• A combination of TSO/E authorized commands and functions such as SYSOUTTRAP. This rule applies only when you invoke the CLIST from the IPCS dialog.3. Invoke the CLIST <i>without</i> using the TSO subcommand to invoke the CLIST, but instead using the TSO subcommand within the CLIST to prefix any TSO/E commands. By using this method, you do not have to use the BLSGSCMD dialog program to invoke the IPCS subcommands. |
| ISPF commands | Invoke the CLIST from the IPCS dialog or through batch mode IPCS if ISPF is active in batch. Do not use the TSO subcommand to invoke the CLIST from the IPCS dialog. The TSO subcommand prevents the CLIST from using ISPF functions. |

See [z/OS MVS IPCS Customization](#) for the BLSGSCMD dialog program.

Controlling Output from REXX EXECs and CLISTs

There are several ways to control the flow of output from an IPCS REXX EXEC or CLIST. Be aware that you can miss error messages by suppressing the output.

FLAG Parameter

The FLAG parameter on the SETDEF subcommand sets the severity level of messages that will appear in the report. This includes messages from the NOTE subcommand. For more information, see the SETDEF subcommand in [z/OS MVS IPCS Commands](#).

NOSUMMARY Parameter

Several subcommands use the NOSUMMARY parameter to control messages that you want to hide from the output. For example, the DROPDUMP subcommand issues the message “All records for 1 dump dropped” unless you specify NOSUMMARY.

Controlling Messages

The FLAG or NOSUMMARY parameters control some, but not all of the messages that can be issued. For example, IPCS issues an error message every time it tries to access an address that is not in the dump. If your CLIST or REXX EXEC can continue processing despite the error, use one of the following techniques to prevent the message from appearing in the output:

- Use the CLIST statement CONTROL NOMSG to prevent messages in IPCS line mode and batch mode. For example, the BLSCSCAN CLIST uses CONTROL NOMSG as follows:

```
CONTROL NOMSG
VERBX ASMDATA TERM NOPRINT
CONTROL MSG
```

- Use the REXX function OUTTRAP to temporarily suppress all output:

```
Call OutTrap "TsoOutputLine.", "*", "NOCONCAT"
"VERBX ASMDATA TERM NOPRINT"
Call OutTrap "OFF"
```

Note: This method does not suppress output when the REXX EXEC runs in the IPCS dialog.

- Use the PRINT and NOTERMINAL parameters to redirect the output to the print data set temporarily. You can specify PRINT and NOTERMINAL on some IPCS subcommands:

```
Address IPCS "WHERE 1000. ASID(X'0001') PRINT NOTERMINAL"
```

For other subcommands use the EVALDEF and SETDEF subcommands to change the defaults, as shown in this example:

```
Address TSO "ALLOCATE FILE(DUMMYOUT) DUMMY"
Address IPCS
"OPEN PRINT (FILE(DUMMYOUT))"
"EVALDEF REXX(CONFIRM(con) PRINT(prt) SOURCE(src)
    TERMINAL(tim) FLAG(flag))" /* Save user's defaults */
"SETDEF PRINT NOTERMINAL"
"EVALUATE 0FF3458. LENGTH(100) REXX(STORAGE(TEMP))"
"SETDEF "con prt src tim "FLAG("flag")" /* Restore defaults */
"CLOSE PRINT"
```

Chapter 7. Using IPCS in Line Mode

You start an IPCS line mode session by entering the IPCS command at the TSO/E READY prompt. After you enter the IPCS command, an “IPCS” prompt replaces the “READY” prompt, indicating that an IPCS line mode session is active. In IPCS line mode, you can enter IPCS subcommands and invoke REXX execs and CLISTS.

This chapter discusses starting and using IPCS in line mode in the following topics:

- [“Specifying Parameters on the IPCS Command” on page 121](#)
- [“Setting Defaults” on page 122](#)
- [“Invoking ISPF under IPCS” on page 122](#)
- [“Starting the IPCS Dialog from IPCS Line Mode” on page 122](#)

Specifying Parameters on the IPCS Command

When you enter the IPCS command, you specify parameters for the IPCS session. These parameters control the defaults for the IPCS print data set and activate the IPCS problem management and data set management subcommands.

Note: These parameters do not control the session defaults specified by the SETDEF subcommand. See [“Setting Session Defaults” on page 15](#) for more information.

The IPCSPRxx parmlib member contains the session parameters. The PR(nn) parameter on the IPCS command controls which set of parameters to use for your IPCS session. You have the following choices when starting an IPCS session:

- If you do not want to use the IPCS problem and data set management subcommands, enter:

```
IPCS NOPARM
```

NOPARM indicates that the IPCS session is for problem analysis only. IPCS does not use a IPCSPRxx parmlib member. NOPARM improves performance when IPCS initializes a dump or trace for processing. However, the problem management and data set management subcommands will not be available.

NOPARM only affects the IPCSPR00 parmlib member. IPCS still uses BLSCECT and all its imbedded parmlib members during your IPCS session.

- If you want to specify defaults for the IPCS print data set, but not use the problem and data set management subcommands, enter:

```
IPCS
```

IPCS starts the session using the parameters in the default parmlib member IPCSPR00 that is shipped with IPCS. If that member is not usable, IPCS diagnoses the errors detected and ends processing of the IPCS command.

- To start the session using an alternate set of parameters located in the parmlib member, IPCSPRnn, enter:

```
IPCS PARM(nn)
```

The nn is a 1- or 2-digit decimal number, which IPCS appends to IPCSPR to form the name of a PARMLIB member. That member, IPCSPRnn, contains the parameters for this session. You usually do not need to specify parameters unless you will use IPCS problem management or the data set management subcommands.

Setting Defaults

You can set session defaults from IPCS line mode by using the SETDEF subcommand. For example, to specify the dump data set named D83DUMP.SYS001SV.SV30036 as the source, enter:

```
SETDEF DSNAME('D83DUMP.SYS001SV.SV30036')
```

The defaults specified with SETDEF can be overridden on many subcommands by specifying the corresponding parameter on that subcommand. The override will be in effect for the subcommand on which it is entered; the override will not change the value specified with SETDEF.

Note: Although you can use SETDEF to establish a dump data set as the current source, you can also use the DEFAULT parameter on the COPYDUMP and OPEN subcommands.

See [z/OS MVS IPCS Commands](#) for a description of the SETDEF subcommand and a list of the default values supplied with IPCS.

Invoking ISPF under IPCS

If you start ISPF from IPCS line mode, as you might in the process of starting the IPCS dialog, enter “ISPF” after the IPCS prompt:

```
ISPF
```

If ISPF is invoked in any other way, such as with a TSO ISPF subcommand, the system might process CLIST variables incorrectly.

In addition, do not include ISPF and IPCS in the TSO/E authorized command table.

Starting the IPCS Dialog from IPCS Line Mode

You can start the IPCS dialog from IPCS line mode. Before you start the IPCS dialog, certain IPCS library concatenations are required. See [z/OS MVS IPCS Customization](#) for information about IPCS libraries required for the IPCS dialog.

Note: Any CLIST or REXX exec library specified by ALTLIB from TSO/E READY will not be available in the IPCS dialog. See the ALTLIB command in [z/OS MVS IPCS Commands](#) for more information.

The following list shows various ways to start the IPCS dialog from IPCS line mode:

- **Directly from IPCS line mode:**

- **Use the ISPSTART command.** From the IPCS prompt, enter:

```
ISPSTART PGM(BLSG) NEWAPPL(BLSG) PASSLIB PARM(PANEL(BLSPPRIM))
```

- **From an ISPF session with IPCS line mode active:**

You can start ISPF from IPCS line mode by entering:

```
ISPF
```

Next, do one of the following:

- **Select the IPCS option.**

If a customized ISPF selection panel provides an option for the IPCS dialog, select that option.

- **Use the ISPF alternate dialog option.**

If an IPCS option is not available on an ISPF selection panel, use the alternate dialog selection option to enter the following:

```
PGM(BLSG) NEWAPPL(BLSG) PASSLIB PARM(PANEL(BLSPPRIM))
```

Note: You can delete the blanks in the previous line to save space.

– **Use the BLSCLIBD CLIST.**

Choose the COMMAND option. At the prompt enter:

```
EX 'SYS1.SBLSCLI0(BLSCLIBD) '
```

Chapter 8. Using IPCS in Batch Mode

IPCS can be used under the TSO/E terminal monitor program (TMP) in a batch job. Consider submitting a batch job when you:

- Perform time-consuming dump analysis with long REXX execs and CLISTs
- Use IPCS subcommands to print selected portions of a dump
- Load or unload system dump data sets from tape or mass storage

Some IPCS subcommands have restrictions when using IPCS in batch mode. These restrictions are documented with the applicable subcommand.

When you submit IPCS batch jobs, the job gets exclusive access to a user dump directory. The reports go the held output queue or IPCS print data set. You cannot use the dump display reporter panel to browse the results. Printing the analysis results is optional. You might want to write the results to an IPCS print data set and browse the results.

Chapter 2, “Starting IPCS,” on page 13 explains the methods for accessing IPCS and Chapter 9, “Printing IPCS Reports,” on page 129 explains how to direct output from an IPCS session. They also give the data sets that must be allocated for any IPCS session. These conventions should be followed when coding the JCL.

Figure 61 on page 126 through Figure 63 on page 127 provide the JCL needed to run IPCS in batch mode. They are examples of how to perform dump analysis, dump formatting, and utility functions respectively.

This chapter contains:

- “Running CLISTs and REXX Execs in Batch Mode” on page 125
- “Running IPCS Subcommands in Batch Mode” on page 126
- “Copying and Clearing Dumps in Batch Mode” on page 127
- “Combining and Processing Traces in Batch Mode” on page 128

Running CLISTs and REXX Execs in Batch Mode

Figure 61 on page 126 gives the JCL needed to run the BLSCSCAN CLIST, which formats a problem screening report for a stand-alone dump. Note the following:

- The batch job allocates the user dump directory, IPCSU1.DUMP.DIR, assuming that an existing dump directory data set is created and initialized.
- IPCSPARM is needed only if you or your installation has its own IPCS exit routines or specifies other IPCS initialization parameters. If IPCSPARM is omitted, the IPCS parameters from SYS1.PARMLIB or the logical parmlib concatenation are dynamically allocated.
- You can substitute %BLSCSCAN with another CLIST or REXX exec. However, ensure that the CLIST or REXX exec is identified through the SYSPROC DD statement.

```

//IPCSJOB JOB 'acctinfo','IPCSU1 output',MSGLEVEL=(1,1),
//          MSGCLASS=A,CLASS=J,NOTIFY=IPCSU1
//* -----
//*
//*   Input: dump in data set 'IPCSU1.DUMP1.DUMP'
//*   Output:
//*     - IPCS dump directory data set for the input dump
//*       (IPCSDDIR DD)
//*     - Formatted output (SYSTSPRT DD)
//*     - TSO/E messages (SYSTSPRT DD)
//*     All of the output will have message identifiers
//*     printed (due to PROFILE MSGID command in SYSTSIN)
//* -----
//IPCS      EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//IPCSDDIR DD DSN=IPCSU1.DUMP.DIR,DISP=(SHR)
//IPCSPARM DD DSN=DEPTNUM.IPCS.PARMLIB,DISP=SHR
//SYSPROC  DD DSN=SYS1.SBLSCLI0,DISP=SHR
//SYSUDUMP DD SYSOUT=A
//SYSTSPRT DD SYSOUT=A
//SYSTSIN  DD *
PROFILE MSGID
IPCS NOPARM
SETDEF DSN('IPCSU1.DUMP1.DUMP') LIST NOCONFIRM
%BLSCSCAN
END
/*

```

Figure 61. Running the BLSCSCAN CLIST in Batch Mode

Note: If you want to browse the IPCS output at a terminal after the batch job has completed, specify message and SYSOUT classes for held output rather than the MSGCLASS=A and SYSOUT=A on the statements in this example.

Running IPCS Subcommands in Batch Mode

Figure 62 on page 127 is an example of running the RSMDATA subcommand in a batch job. Note the following in this example:

- The batch job allocates the dump directory, IPCSU1.DUMP.DIR, assuming an existing dump directory data set.
- The IPCS session parameters are dynamically allocated from the IPCSPRxx parmlib member.
- The output is directed to the IPCS print data set, IPCSU1.RSMDATA.LISTV, and can be viewed using PDF BROWSE.
- The dump data set is allocated with a ddname of DUMP, which was selected arbitrarily for the example. The dump directory associates the DUMP ddname with the data set 'IPCSU1.DUMP1.DUMP' when it is first initialized.
- The SYSTSIN in-stream data contains two DROPDUMP subcommands.
 - The first DROPDUMP subcommand ensures that the user dump directory being used does not contain any information about another dump data set that was identified to IPCS through the same ddname, FILE(DUMP). If another dump data set had been processed through FILE(DUMP) using the same dump directory and this DROPDUMP subcommand were not included, IPCS would detect the attempt to associate the descriptions of two distinct dumps with FILE(DUMP) and would end the processing of the RSMDATA subcommand.
 - The second DROPDUMP subcommand removes the description of the dump processed, assuming that the next time the same dump directory is used, another dump will be of interest. This is not essential if the next job begins with a DROPDUMP operation similar to the one at the start of the job, but it does eliminate records that will not be reused in subsequent IPCS sessions.


```

//IPCSJOB JOB 'acctinfo','IPCSU1 output',MSGLEVEL=(1,1),
//          MSGCLASS=A,CLASS=J,NOTIFY=IPCSU1
//* -----
//*
//*   Input: dump in data set 'IPCSU1.DUMP1.DUMP'
//*   Output:
//*     - IPCS dump directory data set for the input dump
//*       (IPCSDDIR DD)
//*     - Formatted output (SYSTSPRT DD)
//*     - TSO/E messages (SYSTSPRT DD)
//* -----
//IPCS      EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//IPCSDDIR DD DSN=IPCSU1.DUMP.DIR,DISP=(SHR)
//DUMP      DD DSN=IPCSU1.DUMP1.DUMP,DISP=SHR
//IPCSPRNT DD DSN=IPCSU1.RSMDATA.LISTV,DISP=(OLD,KEEP,KEEP)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
IPCS
DROPDUMP DDNAME(DUMP)
RSMDATA REALFRAME ALL STATUS(VRINT,POLLUTE) RANGE(5:86) +
          DDNAME(DUMP) PRINT NOTERMINAL
DROPDUMP DDNAME(DUMP)
END
/*

```

Figure 62. Running a Subcommand in Batch Mode

Copying and Clearing Dumps in Batch Mode

It is sometimes necessary to copy dump tapes to supply another location with a copy of the dump. It is particularly useful to be able to supply a dump tape with an authorized program analysis report (APAR).

There are several methods, such as the IEBGENER utility program, that can be used to copy a dump from one tape to another. Figure 63 on page 127 shows how to use the IPCS COPYDUMP subcommand to copy a dump. Note that this job specifies IPCS NOPARM, to avoid accessing SYS1.PARMLIB.

```

//IPCSJOB JOB 'acctinfo','IPCSU1 output',MSGLEVEL=(1,1),
//          MSGCLASS=A,CLASS=J,NOTIFY=IPCSU1
//IPCS      EXEC PGM=IKJEFT01,REGION=600K
//IPCSDDIR DD DSN=IPCSU1.DUMP.DIR,DISP=(SHR)
//TAPEIN   DD UNIT=TAPE,LABEL=(1,NL),VOL=SER=DMPIN,
//          DCB=(DSORG=PS,RECFM=FBS,BLKSIZE=29120,LRECL=4160)
//TAPEOUT  DD UNIT=TAPE,LABEL=(2,NL),VOL=SER=DMPOUT,
//          DCB=(DSORG=PS,RECFM=FBS,BLKSIZE=29120,LRECL=4160)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
IPCS NOPARM
COPYDUMP INFILE(TAPEIN) OUTFILE(TAPEOUT) NOPRINT NOCONFIRM
END
/*

```

Figure 63. Running the COPYDUMP Subcommand in Batch Mode

Figure 64 on page 127 shows how to copy and clear a SYS1.DUMPnn data set.

```

//IPCSJOB JOB 'acctinfo','IPCSU1 output',MSGLEVEL=(1,1),
//          MSGCLASS=A,CLASS=J,NOTIFY=IPCSU1
//IPCS      EXEC PGM=IKJEFT01,REGION=600K
//IPCSDDIR DD DSN=IPCSU1.DUMP.DIR,DISP=(OLD,KEEP)
//DUMPIN   DD DSN=SYS1.DUMP99,DISP=SHR
//DUMPOUT  DD DSN=SYS1.OFFDMP99,DISP=(OLD,KEEP)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
IPCS NOPARM
COPYDUMP INFILE(DUMPIN) OUTFILE(DUMPOUT) CLEAR NOPRINT NOCONFIRM
END
/*

```

Figure 64. Copying and Clearing a SYS1.DUMPnn Data Set in Batch Mode

See Chapter 11, “Examples of JCL to Print, Copy, and Clear a Dump Data Set,” on page 137 for more examples of JCL to manage SYS1.DUMPnn data sets.

Combining and Processing Traces in Batch Mode

The following example processes GTF traces in batch mode. The two input data sets, IPCSU1.GTF.SYS1.TRACEA and IPCSU1.GTF.SYS1.TRACEB, were created from a GTF trace on a single system. To process them with GTFTRACE, you must first use COPYTRC to combine the traces into a single data set.

```
//IPCSJOB JOB 'acctinfo','IPCSU1 output',MSGLEVEL=(1,1),
//          MSGCLASS=A,CLASS=J,NOTIFY=IPCSU1
//* -----
//*
//* Input: GTF traces in data sets 'IPCSU1.GTF.SYS1.TRACEA'
//*        and 'IPCSU1.GTF.SYS1.TRACEB'
//* Output:
//*   - Combined GTF traces in data set
//*     'IPCSU1.GTF.SYS1.TRACEALL'
//*   - IPCS dump directory data set for the input dump
//*     (IPCSDDIR DD)
//*   - Formatted output (IPCSPRNT DD)
//*   - TSO/E messages (SYSTSPRT DD)
//* -----
//IPCS      EXEC  PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//IPCSDDIR  DD    DSN=IPCSU1.DUMP.DIR,DISP=OLD
//TRACEA    DD    DSN=IPCSU1.GTF.SYS1.TRACEA,DISP=SHR
//TRACEB    DD    DSN=IPCSU1.GTF.SYS1.TRACEB,DISP=SHR
//TRACEALL  DD    DSN=IPCSU1.GTF.SYS1.TRACEALL,DISP=OLD
//IPCSPRNT  DD    DSN=IPCSU1.GTFTRACE.LIST,DISP=(OLD,KEEP,KEEP)
//SYSTSPRT  DD    SYSOUT=*
//SYSTSIN   DD    *
IPCS NOPARM
SETDEF NOCONFIRM PRINT NOTERM
DROPDUMP DDNAME(TRACEA,TRACEB,TRACEALL)
COPYTRC TYPE(GTF) INFILE(TRACEA,TRACEB) OUTFILE(TRACEALL)
GTFTRACE DDNAME(TRACEALL) IO(D34,D0C,ED8,FFF,2A0,2E4)
SETDEF CONFIRM NOPRINT TERM
END
/*
```

This example first uses the SETDEF subcommand to specify NOCONFIRM. NOCONFIRM allows IPCS to process data sets as traces without prompting the user. In case the dump directory IPCSU1.DUMP.DIR is used for both batch and interactive processing of traces, the second SETDEF subcommand restores the default to NOCONFIRM at the end of the job.

Chapter 9. Printing IPCS Reports

This section describes how to print IPCS reports. To print unformatted dump data sets, see [Chapter 11, “Examples of JCL to Print, Copy, and Clear a Dump Data Set,”](#) on page 137.

Print and table of contents data sets

A print data set contains the output produced when you enter IPCS subcommands with the PRINT parameter.

A table of contents (TOC) data set contains the entries produced by the IPCS table of contents exit service when output is written to the print data set. These entries identify major portions of IPCS output as they are written to the print data set.

Both data sets are sequential and contain blocked variable-length records with American National Standards Institute (ANSI) control characters. You should allocate these data sets before you enter an IPCS subcommand that routes its display to the print data set.

Note: You can use the print data set without allocating a TOC data set. The TOC and print data sets should be different for both data sets to contain correct data.

You can control the printer font for the printed output from your print or TOC data sets. See [“Controlling the print font”](#) on page 131.

Allocating the print and table of contents data sets

If you want to allocate the print data set to the DDNAME IPCSPRNT and assign it to SYSOUT class A, enter this TSO/E ALLOCATE command:

```
ALLOCATE DDNAME(IPCSPRNT) SYSOUT(A)
```

Similarly, to allocate the table of contents data set to the DDNAME IPCSTOC and assign it to SYSOUT class A, enter this TSO/E ALLOCATE command:

```
ALLOCATE DDNAME(IPCSTOC) SYSOUT(A)
```

If you want to allocate either data set to something other than SYSOUT, [Table 6 on page 129](#) gives the attributes to use on the TSO/E ALLOCATE command:

| Table 6. Allocation attributes for print and TOC data sets | | |
|--|---|--------------|
| Attribute | Print data set | TOC data set |
| DDNAME or FILE | IPCSPRNT | IPCSTOC |
| DISP | NEW, OLD, or MOD Where applicable, the DISP parameter controls where IPCS writes its output. <ul style="list-style-type: none">• MOD causes IPCS to append its output after any records already in a sequential data set.• OLD and NEW cause IPCS to write its output at the beginning of the data set, replacing any records that are already there. Note: DISP(SHR) is not applicable. | |
| DSORG | PS | |
| RECFM | VBA | |

Table 6. Allocation attributes for print and TOC data sets (continued)

| Attribute | Print data set | TOC data set |
|-----------|---|--|
| LRECL | <ul style="list-style-type: none"> • Minimum value: 83 • Maximum value: 255 • Default value: Value specified in session parameters or 137 | |
| | For the print data set, if LRECL does not satisfy the minimum and maximum conditions, IPCS cancels the PRINT option. | For the TOC data set, if LRECL does not satisfy the minimum and maximum conditions, IPCS makes no TOC entry and continues with the PRINT option. |
| | IBM recommends that LRECL be supplied with all allocations of new IPCSPRNT and IPCSTOC data sets to tape or DASD. LRECL needs to be present to enable data management to calculate a system determined blksize for these data sets. | |
| BLKSIZE | <ul style="list-style-type: none"> • Must be greater than or equal to LRECL + 4. • If you specify none, IPCS uses $4 + (4 * \text{LRECL})$ as BLKSIZE. | |
| | For the print data set, if you specify a BLKSIZE less than LRECL + 4, IPCS cancels the PRINT option. | For the TOC data set, if you specify a BLKSIZE less than LRECL + 4, IPCS makes no TOC entry and continues with the PRINT option. |
| SPACE | You determine the space for the data sets. | |

This example of a TSO/E ALLOCATE command for a print data set uses an attribute list to specify a series of desired attributes:

```
ATTRIB LIST1 DSORG(PS) RECFM(V B A) LRECL(125)
ALLOCATE DDNAME(IPCSPRNT) DATASET(sess7.print) NEW KEEP +
        SPACE(10,5) TRACKS USING(LIST1)
```

Assuming your TSO/E user-prefix is IPCSU1, these commands allocate a new data set named IPCSU1.SESS7.PRINT, assigning to it the attributes specified in LIST1. Note that the attribute list does not specify a BLKSIZE. Specifying LRECL gives the system enough information to calculate a system determined blksize appropriate for the data set and lets IPCS know the width of the intended output medium.

For convenience, you might want to create a TSO/E CLIST to allocate your dump directory, print, and TOC data sets. An example of such a CLIST might be:

```
ALLOCATE DDNAME(IPCSDDIR) DATASET(DEBUG) REUSE
ALLOCATE DDNAME(IPCSPRNT) SYSOUT(A) REUSE
ALLOCATE DDNAME(IPCSTOC) SYSOUT(A) REUSE
```

The FREE command can be used instead of REUSE to ensure that the three data sets are free before allocating them.

Opening and closing the print and table of contents data sets

- If you allocated the print and/or TOC data sets using a DDNAME other than IPCSPRNT or IPCSTOC respectively, use the IPCS OPEN subcommand to inform IPCS of the DDNAME. On the OPEN subcommand, you can also specify the title and time stamp to be placed at the top of each printed page, as well as control the print font for the printed output. For example,

```
OPEN PRINT (FILE(ddname) TITLE ('System In Wait' '12-12-88') CHARS(DUMP))
```

- If you allocated the print and/or TOC data sets using the DDNAME of IPCSPRNT or IPCSTOC respectively, use the IPCS OPEN subcommand to open the print and TOC data sets and, optionally,

to specify the title and the time stamp to be placed at the top of each printed page. If you do not use the IPCS OPEN subcommand, IPCS will open DDNAME IPCSPRNT when the first IPCS subcommand that generates a display to be routed to the print data set is processed.

Any time IPCS attempts to open the IPCSPRNT data set, it will also attempt to OPEN IPCSTOC for a table of contents data set.

- If you did not allocate the print data set, or if IPCS cannot open it, IPCS displays a message informing you that the data set is not allocated. Processing of any subcommand that requests routing to the print data set ends. This does not prevent you from continuing with the IPCS session, but if you want to route output to the data set, you will need to allocate it and reenter the subcommand as originally entered. Otherwise just reenter the subcommand, routing the display to the terminal.

IPCS keeps the data set(s) open until you end the IPCS session or until you use the CLOSE subcommand to explicitly release allocated resources. For example, to close the print and TOC data sets, enter

```
CLOSE PRINT
```

Once the data sets are closed, you can use the TSO/E FREE command. If the data sets are SYSOUT data sets, this will make them eligible to be printed.

Defaults set through the PROFILE subcommand are picked up at the time the print or TOC data set is opened. These defaults remain in effect until the print or TOC data set is closed. Any new defaults set after the print or TOC data set is opened do not take effect until the next print or TOC data set is opened. If you use the PROFILE subcommand to specify new defaults when the print or TOC data set is open, IPCS issues message BLS21086I stating that the defaults you requested will be used when the next print data set or TOC data set is opened.

If you retain the print and TOC data sets, you can reopen them during the session. You can also free these data sets, then allocate and open others in the same session. Each time a print data set is opened, page numbers are reset to page 1. The title text and the time stamp are also reset.

Any time the print data set is closed, the TOC data set will also be closed, if it is open.

Controlling the print font

The default print font for IPCS formats dump output into rows of 16 bytes. IPCS prints all alphanumeric characters and a set of 120 special characters, preserving the mixed-case in messages and displays of dump data.

IPCS offers an option for a denser output format. The CHARS(DUMP) print font for the IBM 3800 Printing Subsystem formats dump output into rows of 64 bytes, four times the default amount. To get this denser format, specify the CHARS(DUMP) print font for a print or TOC data set with a SYSOUT or non-SYSOUT output destination by doing one of the following:

- Specify the CHARS(DUMP) option on the TSO/E ALLOCATE command:

```
ALLOCATE DDNAME(IPCSPRNT) SYSOUT(A) CHARS(DUMP)
```

- Copy the contents to SYSOUT using JCL with the CHARS=DUMP parameter on the output DD statement.

When you open the print and/or TOC data sets, specify the CHARS(DUMP) parameter on the OPEN subcommand.

The CHARS(DUMP) print font does present some disadvantages:

- IPCS translates all lowercase letters to uppercase.
- Many of the 120 special characters are translated to periods.
- When formatting AREA-type data solely to print, underscored characters A through F and 0 through 9 will not display correctly on a 3270 terminal when viewed from PDF BROWSE or other similar browse tool, nor on a printout that specifies a font other than CHARS(DUMP).

Note: To get the default IPCS print font on the 3800, use the default ASIS parameter on the OPEN subcommand and specify the CHARS=GTnn font-control option on the output DD statement.

Printing into partitioned data sets

To get output into a partitioned data set (PDS or PDSE), enter the SETDEF subcommand with the PDS parameter.

After you set parameter PDS using the subcommand SETDEF PDS, messages are routed into a PDS. You can also change the defaults on the IPCS Default Values dialog panel (Option 0 — DEFAULTS) by modifying the following field:

```
Message Routing ==> NOPRINT TERMINAL PDS
```

Using the PDS value routes and transmits the output messages of the IPCS subcommand to the appropriate PDS member. The following rules for routing output apply:

- During the period between issuing “SETDEF PDS” and “SETDEF NOPDS”, the output messages of any IPCS subcommand are routed into a PDS member with a name equivalent to the name of the subcommand.
- If the same subcommand is used several times during an IPCS session, IPCS appends the new output to the previous output in the same member.

Note:

1. When a subcommand is invoked explicitly in an IPCS session or from a REXX procedure, any output messages from IPCS subcommands are routed to the PDS. Output messages of CLIST procedures are not routed to the PDS because they run asynchronously relative to IPCS.
2. When the VERBEXIT or VERBX subcommand is invoked in an IPCS session, its output messages are routed into a PDS member with a name equivalent to the name of the verb exit routine.

Allocating the partitioned data set for routing messages

The user must create the PDS before trying to output subcommand messages to it.

The name of the partitioned data set can be included in the JCL used to start IPCS in batch mode by means of DDNAME IPCSPDS, for example:

```
//IPCSPDS DD DSN=IPCSU1.SESS7.PDS,DISP=SHR
```

If IPCS is started using the TSO command IPCS in line mode or in full-screen mode, you can allocate DDNAME IPCSPDS with the TSO command ALLOCATE, for example:

```
ALLOC DDNAME(IPCSPDS) DSNAME('IPCSU1.SESS7.PDS') SHR
```

Using allocation attributes for print partitioned data sets

Although you can use PDS or PDSE organization, using PDSE (DSNTYPE=LIBRARY) provides for best performance and avoids any fragmentation of the PDS space.

Use the FREE command after termination of any IPCS subcommand to free the allocated partitioned data set, for example:

```
FREE DDNAME(IPCSPDS)
```

Table 7 on page 132 shows the attributes to use when allocating a partitioned data set for routing IPCS messages.

| Table 7. Allocation attributes for print partitioned data sets | |
|--|----------------------------|
| Attribute | Print partitioned data set |
| DDNAME or FILE | IPCSPDS |
| DISP | NEW, SHR or OLD |

| Table 7. Allocation attributes for print partitioned data sets (continued) | |
|--|---|
| Attribute | Print partitioned data set |
| DSORG | PO |
| DSNTYPE | LIBRARY or no |
| RECFM | VBA |
| LRECL | Minimum value: 83 Maximum value: 255 Default value: the value specified in session parameters or 137 For the PDS, if LRECL does not satisfy the minimum and maximum conditions, IPCS cancels the PDS option. |
| BLKSIZE | Must be greater than or equal to LRECL + 4 If you specify none, IPCS uses 4 + (4 * LRECL) as BLKSIZE For the PDS, if you specify a BLKSIZE less than LRECL + 4, IPCS cancels the PDS option. |
| SPACE | You determine the space for the partitioned data sets. |

Setting print data set report defaults

IPCS provides a subcommand, PROFILE, that allows you to establish a preferred line size and preferred lines per printed page for reports generated under IPCS.

Like the SETDEF-defined parameters, IBM supplies PROFILE-defined defaults. See the PROFILE subcommand in [z/OS MVS IPCS Commands](#) for the defaults.

The defaults specified with PROFILE are recorded in the dump directory and remain in effect until changed. You can enter the PROFILE subcommand at any time during an IPCS session to view the default values. To establish a new default, enter the PROFILE subcommand with the parameter that specifies the default you want.

Note, however, new default settings specified while a print or table of contents data set is already open do not take effect until the print or table of contents data set is reopened. Except for NOPAGESIZE the newly established default is used for both the current session and any subsequent sessions in which the same dump directory is used. NOPAGESIZE does not become effective until the beginning of the next IPCS session. Unlike the SETDEF subcommand, you cannot override the PROFILE defaults on individual IPCS subcommands. The defaults can be changed only by entering the PROFILE subcommand. For example, to limit reports to a line length of 78 characters with 30 lines of data per page, enter:

```
PROFILE LINESIZE(78) PAGESIZE(30)
```

Parameters on the PROFILE subcommand

The following describes the parameters used to change the print data set report defaults.

LINESIZE(nnn) | NOLINESIZE

Use the LINESIZE parameter to limit the width of variable-width reports. The minimum line size is 78 and the maximum is 250.

If variable-width reports are sent to any medium that is narrower than nnn characters, IPCS limits the output lines from the report to the width of the medium or 78 characters, whichever is larger.

NOLINESIZE specifies that variable-length reports use the full width of the medium to which they are written.

Obtaining the LINESIZE

IPCS obtains the width of the terminal through the ISPF PQUERY service when the reports are generated under ISPF and through the TSO/E GTSIZE service otherwise. TSO/E display terminals are regarded as having lines two characters shorter than their physical line sizes to allow for the use of screen control characters.

IPCS obtains the width of the print data set output from one of three sources:

1. A nonzero LRECL value made available by the MVS OPEN service when the print data set is prepared for use. This is the preferred source.
2. A LINELENGTH specified in the IPCSPRxx parmlib member for the IPCS session. LINELENGTH is five characters greater than line size to allow for an ASA control character plus control information for variable-length records. This is the preferred source if a zero LRECL value is detected.
3. The IPCS default of 132 characters. This is used when neither of the preceding is provided.

Note: Some reports generated under IPCS have fixed line lengths as long as 132 characters. The LINESIZE value specified on the PROFILE subcommand has no effect on such reports.

PAGESIZE(nnn) | NOPAGESIZE

Use PAGESIZE to specify the number of lines per page for the IPCS print output data set. The maximum page size is $2^{31}-1$.

IBM recommends that you use a PAGESIZE that corresponds with the number of lines that will fit on the forms typically used at your installation. IPCS can only generate normal, ascending page numbers if the printed output consumes less than 2^{32} lines of output medium. If you use a large PAGESIZE, the page number will wrap back to zero once the maximum is reached.

Obtaining the PAGESIZE

IPCS obtains the number of lines per page for the IPCS print output data set by checking the following in order:

1. The PAGESIZE specified on the PROFILE subcommand.
2. The PAGESIZE specified in the session parameters member for the IPCS session. (If PROFILE NOPAGESIZE is in effect, IPCS checks here first.)
3. When neither of the preceding is available, IPCS uses the default of 60 lines per page.

Note: Entering PROFILE NOPAGESIZE does not alter the default for the current IPCS session. It becomes effective at the beginning of the next IPCS session.

STACK(DUPLICATES | NODUPLICATES)

Use STACK to control duplication of pointer stack entries. Entering PROFILE STACK(DUPLICATES) allows duplication of stack entries.

Entering PROFILE STACK(NODUPLICATES) suppresses duplicate stack entries, but does not affect entries generated by any of the following:

- the EQUATE and RUNCHAIN subcommands
- the INSERT and REPEAT dialog line commands
- editing the pointer stack from the pointer panel

STACK affects the current IPCS session, and any future sessions that use the same dump directory.

Chapter 10. Scenario to Obtain Spin Loop Diagnostic Information

This scenario uses IPCS to gather useful debugging information pertaining to a spin loop problem on a tightly coupled multi processor (MP). It is presented as an example, illustrating how one can use IPCS online to analyze system problems.

System Environment — There is no message movement on the console. The system issues message IEE331A, which says:

```
PROCESSOR (1) IS IN AN EXCESSIVE DISABLED SPIN LOOP  
WAITING FOR INTERSECT RELEASE  
REPLY U TO CONTINUE SPIN,  
OR STOP PROCESSOR (2) AND REPLY ACR.  
(AFTER STOPPING THE PROCESSOR, DO NOT START IT)
```

The operator attempted recovery but recovery failed. The operator then took a stand-alone dump and copied the dump to DASD specifying the data set name of 'stand.alone.spin'.

Using the IPCS dialog, a diagnostician might perform the following steps:

1. Access IPCS and the full-screen dialog by doing the steps outlined in [“Starting IPCS” on page 13](#).
2. Make 'stand.alone.spin' the current dump source. Select option 0 from the IPCS Primary Option Menu and enter the data set name in the source field.
3. Press PF3 to exit.
4. Verify that this data set is the correct dump by checking its title. To do this, select option 4 from the IPCS Primary Option Menu and enter:

```
list title
```

5. Press PF3 to exit.
6. If a copy of the message or console log is unavailable, find the message (which indicates the processor that was in the spin loop and the reason why) on the delayed issue queue. Here's how:
 - a. Get the report that lists branch-entry messages by entering:
7. Press PF3 to exit.
8. Obtain the failing CPU's program status word (PSW) and registers and the module that was running by entering:

```
verbexit mtrace
```

```
find branch-entry
```

- c. Search through the list of messages for the one that indicates the processor was in a spin loop.

```
status cpu worksheet regs
```

9. Press PF3 to exit.
10. At this point in the debugging process, you can contact the IBM Support Center and provide the failing CPU, the PSW's address, the registers, and the module name and its displacement.

Chapter 11. Examples of JCL to Print, Copy, and Clear a Dump Data Set

The following samples of JCL provide functions formerly provided by AMDPRDMP.

Sample of JCL for Printing Dumps Using IPCS

The following JCL prints a dump, using IPCS, from the system dump data set:

```
//SUMDUMP JOB
//S EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//DUMP DD DSN=SYS1.DUMP02,DISP=SHR
//SYSPROC DD DSN=SYS1.SBLSCLI0,DISP=SHR
//SYSTSPRT DD SYSOUT=A
//IPCSPRNT DD SYSOUT=A
//IPCSTOC DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSTSIN DD *
  PROFILE MSGID
  %BLSCDDIR VOLUME (vsampk)
  IPCS NOPARM
  SETDEF DSN('SYS1.DUMP02') NOCONFIRM
  VERBEXIT SUMDUMP
/*
```

Figure 65. Example JCL to Print a Dump with IPCS

- vsampk is any VSAM volume.
- You may also replace VERBEXIT SUMDUMP with SUMMARY FORMAT, or %BLSCBSVB, or other IPCS commands, or CLISTS.

Sample of JCL for Printing Dumps into Print and Partitioned Data Sets

The following JCL prints a dump, using IPCS, from the system dump data set into Print and Partitioned Data Sets:

```
//SUMDUMP2 JOB
//GTFPRNT EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//DUMP DD DSN=SYS1.DUMP02,DISP=SHR
//SYSPROC DD DSN=SYS1.SBLSCLI0,DISP=SHR
//SYSTSPRT DD SYSOUT=A
//IPCSDDIR DD DSN=SYS4.DDIR,DISP=SHR
//IPCSPRNT DD SYSOUT=A
//IPCSTOC DD SYSOUT=A
//SYSTSIN DD *
  ALLOC DDNAME(IPCSPDS) DSNAME('IPCSU1.SESS7.PDS') SHR
  IPCSDDIR 'SYS4.DDIR'
  IPCS NOPARM
  SETDEF DDNAME(DUMP) LIST NOCONFIRM
  SETDEF PRINT PDS
  VERBX SUMDUMP
  SETDEF NOPRINT NOPDS
  FREE DDNAME(IPCSPDS)
  END
/*
```

Figure 66. Example JCL to Print a Dump with IPCS into Print and Partitioned Data Sets

Example of JCL to Print, Copy, and Clear a Dump Data Set

Many users of PRDMP previously ran a job that would do the following:

- Perform specified analysis on the SYS1.DUMPnn data set.
- Copy the dump to either another data set, a generation data group (GDG) data set or a tape.
- Clear the SYS1.DUMPnn source data set so that it is available to the system for the next dump request.

The following JCL shows an example of how to accomplish the above functions using IPCS:

```
//IPCSJOB JOB
//*
//* SUBMIT THIS JOB TO DELETE THE DUMP DIRECTORY,
//* INVOKE IPCS, DO WHATEVER ANALYSIS IS WANTED,
//* COPY THE SOURCE DUMP DATA SET TO ANOTHER
//* DUMP DATA SET, AND CLEAR THE SOURCE DUMP DATA SET.
//*
//* THE INPUT SYS1.DUMPNN DATA SET MUST BE ALLOCATED SHR
//* TO AVOID CONFLICTS WITH THE DUMPSRV ADDRESS SPACE.
//*
//* MODIFY THE DUMPOUT DD STATEMENT TO REFLECT WHERE YOU WANT THE
//* DUMP COPIED (GDG, OTHER DATASET, OR TAPE).
//*
//IPCS      EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//SYSPROC   DD DSN=SYS1.SBLSCLI0,DISP=SHR
//IPCSDUMP  DD DSN=SYS1.DUMP00,DISP=SHR
//DUMPOUT   DD DSN=GDG.DATA.SET.NAME,DISP=SHR
//SYSUDUMP  DD SYSOUT=A
//IPCSPRNT  DD SYSOUT=A
//IPCSTOC   DD SYSOUT=A
//SYSTSPRT  DD SYSOUT=A
//SYSTSIN   DD *
/* THE DELETE STATEMENT WILL DEFAULT TO THE USERID OF THE */
/* BATCH JOB. YOU CAN FULLY QUALIFY IT IF YOU WANT TO. */
DELETE (DDIR) PURGE CLUSTER
/* THE BLSCDDIR CLIST WILL DEFAULT TO VOLUME VSAM01. */
/* OVERRIDE THE VOLUME WITH WHERE YOU WANT THE DUMP DIRECTORY */
/* ALLOCATED. */
BLSCDDIR VOLUME(VSAM11)
IPCS NOPARM
SETDEF DD(IPCSDUMP) LIST NOCONFIRM
/* AT THIS POINT REPLACE THE LIST 0 COMMAND WITH WHATEVER */
/* ANALYSIS YOU NORMALLY RUN AGAINST THE DUMP FOR PRELIMINARY */
/* ANALYSIS. CONSIDER CLIST BLSCBSVA. */
LIST 0
/* OFFLOAD THE SYS1.DUMPNN DATA SET AND THEN CLEAR IT. */
COPYDUMP INFILE(IPCSDUMP) OUTFILE(DUMP00) NOPRINT NOCONFIRM CLEAR
END
/*
```

Figure 67. Example JCL to Run IPCS for Printing, Copying and Clearing a Dump Data Set

Example of a Procedure to Print, Copy, and Clear a Dump Data Set

The following example performs similar actions to the JOB in the previous example.

```

//IPCS      PROC CLIST=PRSVCDMP,DUMP=,OUTDUMP=,CLEAR=NOCLEAR
//IEFPROC   EXEC PGM=IKJEFT01,REGION=4M,DYNAMNBR=10,
// PARM=('%&CLIST.','&DUMP.','&OUTFILE','&CLEAR')
//*
//*          INPUT PARAMETERS
//*
//* 1. CLIST=name - The name of the CLIST that is to be executed
//* 2. DUMP=name - The name of the dump data set to process
//* 3. OUTDUMP=name - The name of the DASD data set to receive a
//*                  copy of the dump
//* 4. CLEAR=option - Where option is CLEAR or NOCLEAR. This controls
//*                  whether the source dump data set is cleared.
//*
//*          INPUT DATA SETS
//*
//IPCSDUMP DD DSN=&DUMP,DISP=SHR          DUMP OR TRACE DATA SET
//IPCSPARM DD DSN=SYS1.PARMLIB,DISP=SHR    IPCS PARMLIB MEMBERS
//SYSPROC DD DSN=SYS1.SBLSCLI0,DISP=SHR    CLIST PROCEDURES
//SYSTSIN DD DUMMY,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)  TSO/E COMMANDS
//*
//*          OUTPUT DATA SET
//*
//OUTFILE DD DSN=&OUTDUMP,DISP=SHR          OUTPUT DATA SET
//*
//*          FORMATTED OUTPUT
//*
//SYTSPT DD SYSOUT=A                      BATCH TSO/E SESSION LOG
//IPCSTOC DD SYSOUT=A                      PRINT FILE TABLE OF CONTENTS
//IPCSPRT DD SYSOUT=A                      PRINT FILE

```

Figure 68. IPCS Procedure to Print, Copy and Clear a Dump Data Set

You can invoke the above IPCS procedure with JCL or from an MVS operator START command.

```

//IPCSJOB JOB
//RUN EXEC IPCS,DUMP='SYS1.DUMP99',OUTDUMP='SYS1.OFFDMP99',
// CLEAR='CLEAR'

```

Figure 69. JCL to Invoke an IPCS Procedure to Print, Copy and Clear a Dump Data Set

The above JCL will process data set SYS1.DUMP99. The analysis is defined in CLIST PRSVCDMP. After performing the analysis, the dump is copied to data set SYS1.OFFDMP99. If the copy of the dump is successful, the SYS1.DUMP99 data set is cleared.

```

START IPCS,DUMP='SYS1.DUMP99',OUTDUMP='SYS1.OFFDMP99',CLEAR=CLEAR

```

Figure 70. START Command Example to Print, Copy and Clear a Dump

The above operator START command will perform the same processing as the JCL example.

The CLIST used in the example is called PRSVCDMP. It was created by modifying CLIST BLSCBSVA with additional parameters to control the copying of the dump and an option to CLEAR the source dump data set.

```

PROC 3 DSN OUTFILE CLEAR
/*-----*/
/* CLIST NAME = PRSVCDMP */
/* INPUT = 1. Dump data set name to process */
/*          2. Name of DD statement as target of offload */
/*          3. Control for whether source dump should be */
/*             cleared. Specify CLEAR or NOCLEAR. */
/* OUTPUT = Dump analysis is generated and dump is copied */
/*           and cleared if requested. */
/*-----*/
CONTROL LIST END(ENDO) NOFLUSH ASIS
IPCS NOPARM
IF &LASTCC>8 THEN EXIT CODE(&MAXCC)
/*-----*/
/* Save default environment current when PRSVCDMP started */
/*-----*/
EVALDEF CLIST(CONFIRM(CON) PRINT(PRT) SOURCE(SRC) TERMINAL(TRM+
) FLAG(FLG))
IF &LASTCC>8 THEN EXIT CODE(&MAXCC)
/*-----*/
/* Establish a default environment during PRSVCDMP operation */
/*-----*/
SETDEF DSNNAME('&DSN') NOCONFIRM NOPRINT TERMINAL FLAG(ERROR)

IF &LASTCC>8 THEN EXIT CODE(&MAXCC)

/*-----*/
/* Delete any residual dump description associated with the */
/* data set name of the new dump. */
/*-----*/
CONTROL NOMSG
DROPDUMP
IF &LASTCC>8 THEN DO
    CONTROL MSG
    GOTO EXITSETD
    ENDO
CONTROL MSG

/*-----*/
/* Prepare the dump for analysis */
/*-----*/
OPEN DSNNAME
IF &LASTCC>8 THEN GOTO EXITSETD
/*-----*/
/* Route analysis results regarding the dump to IPCSPRNT */
/*-----*/
OPEN PRINT
IF &LASTCC>8 THEN GOTO EXITSETD
SETDEF PRINT NOTERMINAL FLAG(ERROR)
IF &LASTCC<=8 THEN DO
    /*-----*/
    /* Generate an analysis report for the SVC dump */
    /*-----*/
    %BLSCBSVB LIST
    /*-----*/
    /* Offload the SVC dump and clear the input data set */
    /*-----*/
    COPYDUMP INDSNAME('&DSN') OUTFILE(&OUTFILE) &CLEAR
    ENDO
    /*-----*/
    /* Restore default environment current when PRSVCDMP started */
    /*-----*/
EXITSETD: SETDEF &CON &PRT &SRC &TRM FLAG(&FLG)
END

```

Figure 71. IPCS CLIST Used to Print, Copy and Clear a Dump Data Set

The IPCS procedure in Figure 68 on page 139 and the PRSVCDMP CLIST in Figure 71 on page 140 work together to provide the function of printing, copying and clearing a dump data set. You should use the following steps in tailoring similar procedures and CLISTs to perform their normal dump processing tasks.

1. Create a procedure in SYS1.PROCLIB. IPCS provides a procedure called BLSJIPCS which has an alias of IPCS in SYS1.PROCLIB. This provides the base for the tailored procedure. You should name the procedure something other than IPCS to avoid conflict with the one provided with MVS. Tailor the procedure to look like the one in Figure 68 on page 139.

2. Tailor the OUTFILE DD statement to reflect where you want the SVC dumps to be copied. This can be modified to define a tape for receiving dumps.
3. Tailor the defaults for the procedure to save operator keystrokes. For example, if the SYS1.DUMPnn data sets are always to be cleared, change the “CLEAR=NOCLEAR” to “CLEAR=CLEAR”. The OUTDUMP parameter can be eliminated if the OUTFILE DD statement always goes to a non-label (NL) tape. The DUMP parameter could be modified to require the operator to specify only the last 2 digits of the dump data set name. The ideal setup would be a procedure which can be invoked as S IPCS,DUMP=01 to process a dump in SYS1.DUMP01.
4. For the formatted output, change the SYSOUT=A to the desired print class.
5. When this procedure is run as a START command, it will use or create a dump directory called SYS1.DDIR. Before using this dump directory, the CLIST will always clear out any information saved for the previous dump of the same name. This prevents the SYS1.DDIR dump directory from running out of space.
6. The SYSPROC DD statement identifies the library or libraries where CLISTs are located. This procedure and CLIST combination require the following CLISTs to be in a data set in the SYSPROC concatenation:

PRSVCDMP

Customer-tailored CLIST as shown in [Figure 71 on page 140](#).

BLSCDDIR

CLIST provided by IPCS to define dump directories. This CLIST is automatically invoked by the IPCS command. The CLIST makes sure that the job or user has a dump directory allocated. If one is not allocated, the CLIST will allocate or define one for the user. This CLIST should be modified to identify the DASD volume where the installation wants the VSAM dump directory allocated. The default volume is VSAM01.

BLSCBSVB

CLIST provided by IPCS to perform dump analysis. The system programmer should examine this CLIST to determine if it is providing the desired initial analysis. Either modify this CLIST or create an installation tailored CLIST to replace it. Be sure you invoke the desired CLIST from the PRSVCDMP CLIST.

Chapter 12. Managing Problems and Data Sets

This section describes how to use IPCS subcommands to manage problems and their associated data sets. For detailed information about the individual subcommands, see the appendix on problem management subcommands in [z/OS MVS IPCS Commands](#).

However, IBM makes the following recommendations:

- If you are not currently using a problem management tool to report and track problems detected in your organization's data processing environment, use the IBM Information/Management facility instead of the IPCS problem management subcommands.
- If you currently use the IPCS problem management subcommands to report and track problems, consider migrating to the IBM Information/Management facility.

IBM does not plan to enhance or update the IPCS problem management subcommands, and you should be aware that these subcommands do not support all of the current codes for authorized program analysis reports (APARs) and program temporary fixes (PTFs).

For more information about the IBM Information/Management facility, see *Information/Management Library: Problem, Change, and Configuration Management User's Guide*

Managing Problems

Managing problems consists of adding them to the IPCS problem directory, specifying their attributes, modifying their attributes, listing them, and deleting them from the problem directory.

Problems defined to IPCS must be owned by someone. A problem owner usually is the person responsible for the problem and its solution. You specify the owner when you add the problem to the IPCS problem directory. You can specify the owner's TSO/E userid or let it default to the TSO/E user ID you are using when you add the problem.

Restrictions on processing problems are imposed through ownership of the problems. You can modify or delete a problem only if you own it. Optionally, your installation can designate a person as having administrative authority and a person (possibly the same one) as having delete authority. See the description of the IPCSPRxx member of SYS1.PARMLIB in [z/OS MVS Initialization and Tuning Reference](#).

The person with administrative authority has the same authority as a problem owner but has that authority over all problems in the problem directory. The person with delete authority is the only person who can delete problems from the problem directory. If your installation designates someone with delete authority, neither problem owners nor the person with administrative authority can delete problems.

If no one at your installation has administrative authority or delete authority, the owner of a problem is the only person who can modify the problem's attributes and delete it.

Note: The concepts of problem ownership, administrative authority, and delete authority are disciplines established by IPCS. Since IPCS runs as a problem program (non-authorized), these disciplines cannot be fully enforced. Access to data sets, however, is controlled through conventional system security facilities.

Allocating a Problem Directory

The IPCS problem directory contains the list of problems defined at your IPCS installation. For each problem defined, it contains the problem attributes and the names of data sets associated with that problem.

The problem directory is a VSAM indexed cluster. It is created with Access Method Services. The problem directory must be created and initialized before the first IPCS session at your installation.

The name of the problem directory must be specified in the IPCS session parameters member. If the session parameters do not specify the problem directory data set name, the IPCS command abends. You

them with its own data set directory. You should always use the same problem directory with the same data set directory since records in each data set point to records in the other.

Problem Attributes

A problem's attributes give a complete description of it. The attributes for a problem are:

Problem identifier

Problem status

Problem occurrence date

FIX status

Problem occurrence time

IBM status

Problem reported date

PTF status

Problem reported time

APAR identifier

Group

PTF identifier

Owner

FIX identifier

System

Abstract

Component

Description

Severity

User

IPCS uses the problem identifier to uniquely identify a problem. The problem identifier is composed of a three-character prefix and a five-decimal-digit suffix. The prefix is used to differentiate between problems with similar numbers in different problem directories. It is specified in your session parameters member, but you never specify it on an IPCS subcommand. When IPCS lists problems, it displays their complete problem identifiers, including prefixes. The prefix is not stored in the problem directory and can be changed without affecting previous work. IPCS assigns the suffix when you add a problem. After the problem identifier is stored, you cannot change it. You specify the suffix on any IPCS subcommand needing identification of the problem.

The date and time of problem occurrence allow you to specify when a problem occurred. IPCS sets the date and time of problem reporting when you add the problem to the problem directory. You can modify the problem occurrence attributes but you cannot modify the problem reported attributes.

The group and owner identify both the group and the individual responsible for tracking and resolving the problem. The group is the department or organization name or any other identification you want to use for your installation. You can change the group at any time. You can specify an owner for the problem by specifying a TSO/E userid, or IPCS supplies your TSO/E userid as the default. The owner is significant because certain IPCS subcommands can be run against a problem only by its owner. As owner of a problem, you can modify its attributes and, unless restricted, delete it. You can transfer ownership of a problem by modifying this field.

The system and component attributes identify the environment of the problem. The system identifier refers to the hardware or software system on which the problem occurred. The component identifier refers to the software component you suspect has caused the failure.

The severity attribute allows you to specify a numerical estimation of the severity of the problem, consistent with standard IBM APAR severity codes.

The status and identifier attributes allow you to provide information about the status and resolution of the problem. The status attributes -- PROBLEM, FIX, IBM, and PTF -- allow you to indicate the status of the

problem and the status of its resolution. You can indicate the status from your point of view (PROBLEM and FIX status) and from IBM's point of view (IBM and PTF status). For each status attribute, IPCS records the date and time of the last change to that attribute.

The identifier attributes -- APAR, PTF, and FIX -- allow you to indicate the APAR for the problem and the PTF or FIX that might resolve it or that does resolve it.

The abstract and description allow you to describe the problem, first briefly in the abstract, then at length in the description.

The user data attribute allows you to specify additional information about the problem. It is defined by the user or by the installation.

Managing Data Sets Associated with Problems

Managing data sets associated with IPCS problems consists of associating them with a problem, specifying and modifying their attributes, listing their attributes, and dissociating them from problems. Any kind of data set that you find meaningful can be associated with a problem. The problem must be defined to IPCS before you can associate a data set with it.

Unlike problem ownership, IPCS does not establish data set ownership. Restrictions on processing data sets are imposed through the ownership of the problems the data sets are associated with. You can associate a data set with a problem whether or not you own that problem. You can list the attributes of a data set whether or not you own the problem or problems it is associated with.

You can modify a data set's attributes only if you own one of the problems it is associated with. You can dissociate a data set from a problem only if you have administrative authority or if you own the problem.

Allocating a Data Set Directory

The IPCS data set directory contains the list of data sets associated with the problems defined in the corresponding problem directory. The IPCS data set directory contains, for each data set associated with a problem, the attributes of the data set and the identifier of the problem or problems the data set is associated with.

The data set directory is a VSAM indexed cluster. It is created with Access Method Services. The data set directory must be created and initialized before the first IPCS session at your installation.

The name of the data set directory must be specified in the IPCS session parameters. If the session parameters do not specify the data set directory name, the IPCS command abends. You do not allocate the data set directory for an IPCS session. IPCS dynamically allocates it during IPCS initialization.

Once allocated, IPCS opens and closes the data set as needed but leaves it allocated for the duration of the session. IPCS allocates the data set directory with a disposition of SHR and serializes access to it.

The characteristics of this VSAM key-sequenced cluster are:

```
INDEXED KEYS(61 0) RECORDSIZE(95 144) SHAREOPTIONS(1)
```

The DASD space required by this cluster is variable. Assuming a control interval size of 4096, approximately 57 data set record groups fit on a 3330 track, assuming each data set is associated with 2 problems.

The following example uses the DEFINE command to create a data set directory cluster. Replace the name with one meaningful to you.

```
define cluster(name('ipcs.data.set.directry') volumes(111111) +
  indexed keys(61 0) cylinders(3 3) recordsize(95 144) +
  controlintervalsize(4096) shareoptions(1) )
```

Assuming that 111111 is a 3330 volume, the space allocation in this example is sufficient for approximately 3000 data sets fitting the description above.

The data set directory is initialized by placing a seed record into it. The following procedure uses TSO/E and Access Method Services during a TSO/E session for the initialization and assumes that the data set directory has been previously defined and allocated using Access Method Services:

1. Use TSO/E EDIT to create a new data set and place a single 80-character record in it. This is the seed record.

```
edit 'dsdseed' data new nonum
```

Create the following record in character format:

```

                                     5
1                                     3
000000000000 ... 000000000000000000000000DUMYRECD0000
<----- 53 zeros ----->
```

2. Save the data set.
3. Allocate the VSAM cluster previously defined as the data set directory:

```
alloc fi(b) ds('ipcs.data.set.directry') old
```

where 'IPCS.DATA.SET.DIRECTRY' is the name of the data set directory specified in your session parameters member.

4. Use the Access Method Services REPRO command to put the seed record into the data set directory:

```
repro ids('dsdseed') outfile(b)
```

where 'outfile(b)' specifies the data set directory, allocated in step 3.

5. Free the data set directory:

```
free fi(b)
```

You must define multiple data set directories if you define multiple problem directories. Each data set directory must be paired with its own problem directory since records in each data set point to records in the other.

Attributes of Data Sets

There are three attributes for any data set associated with a problem:

- Description
- Management
- Type

The description is a brief explanation of the data set's contents. The management attribute specifies whether or not IPCS attempts to scratch the data set when it is no longer associated with any problem. The type attribute identifies the kind and format of data in the data set.

When you associate a data set with a problem you can let IPCS manage it. If you let IPCS manage a data set, IPCS will, under certain circumstances, scratch the data set when it is no longer associated with any problem in the problem directory.

IPCS scratches and uncatalogs a managed data set when the following three conditions are met:

- It is cataloged.
- It is no longer associated with any problem in the IPCS problem directory.
- Its name does not begin with 'SYSn.', where n is a decimal number from 0 through 9.

IPCS never scratches a partitioned data set member.

If the data set's name is SYS1.DUMPnn, where nn is a decimal number from 00 through 99, IPCS initializes the data set for reuse rather than scratches it. In order for IPCS to initialize a SYS1.DUMPnn data set, it must be cataloged and be on direct access storage.

Data sets that may be associated with problems are any data sets that you consider valuable or useful in solving a problem. Such data sets can be dumps, assembler and compiler listings, utility and service aid outputs, etc.

These data sets are optional. It is possible to define a problem to IPCS and record its status and resolution without using a single dump or listing or any other data set except those used by IPCS itself.

IPCS classification supports three types of data sets associated with problems:

- Dump data sets
- Print data sets
- User-defined data sets

- **Dump Data Sets**

Dump data sets contain information that has been retained unaltered since an incident occurred. They may contain an image of system storage generated by the AMASADMP program at the time of an unscheduled re-IPL or generated by the SDUMP macro during system operation.

- **Print Data Sets**

Print data sets contain printable data related to a problem. These data sets may contain formatted dumps or the output of utilities, service aids, assemblers, compilers, installation-written routines, etc.

- **User-Defined Data Sets**

User-defined data sets are data sets associated with a problem that do not qualify as dumps or print data sets. They can have any data set organization and can contain any data related to a problem.

Chapter 13. Managing Dump and Trace Data Sets

IPCS processes data sets that reside on DASD or on magnetic tape. These data sets can be accessed with the basic direct access method (BDAM), basic partitioned access method (BPAM), basic sequential access method (BSAM), or VSAM. The selection of an access method depends on the attributes stored in the job file control block (JFCB), tape label, or the data set control blocks.

Performance Consideration

Many IPCS subcommands require random access to retrieve storage from a dump data set. If the dump data set resides on magnetic tape, you should move the dump data set to DASD before processing the dump. Processing the dump from a tape data set may take considerably longer.

Considerations for Managing the Data Sets

The following considerations apply to common procedures used to manage dump and trace data sets for IPCS source processing.

Data Sets with Sequential or Direct Organization

There are a number of methods you can use to manage system dump or other data sets that have the following characteristics:

- DSORG=PS for sequential organization or DSORG=DA for direct organization
- RECFM=F or RECFM=FBS for fixed-length records.

You can manage such data sets by using various methods, including:

- Moving the data set using the Hierarchical Storage Manager (HSM)
- Moving the dump data from one magnetic tape to another magnetic tape
- Moving the dump data from a magnetic tape to a DASD volume or from a DASD volume to a magnetic tape
- Moving the data set from one type of DASD to another (for example, from a 3350 to a 3380).
- Consolidating the data set extents using Data Facility Data Set Services (DFDSS).

All of the preceding operations preserve the data set description in the dump directory.

For data sets that do not have these characteristics, as long as the relative track positions (TTR) of the physical blocks are not altered, you can perform operations that include the following:

- Moving the data set with the HSM
- Moving the dump data from one magnetic tape to another magnetic tape
- Consolidating the data set extents using DFDSS.

Avoid moving the data set to a different type of DASD, because this usually obsoletes the data set description in the dump directory.

Data Sets with Partitioned Organization

Avoid moving a partitioned data set (PDS) with the HSM, eliminating fragmentation using Data Facility Data Set Services, and compressing using IEBCOPY. These activities usually obsolete the data set description in the dump directory. You should avoid using similar tools for the same reason.

Dump Data Set Initialization

There are certain points concerning dump data set initialization that you should keep in mind:

- System dumps stored as members of a PDS cannot be initialized using IPCS system conventions.
- For IPCS to initialize a SYS1.DUMPnn data set, the data set should not be in the system's list of SYS1.DUMP data sets. Use the IPCS subcommand COPYDUMP CLEAR to clear the data set or the operator command DUMPDS DEL to both clear and remove the data set from the system's list. Clearing the SYS1.DUMP data sets periodically will help to avoid the situation where the dump of an important problem cannot be written because all of the SYS1.DUMPnn data sets are full.

Reblocking Dump Data Sets

Reblocking a dump data set obsoletes the dump's description in the dump directory. If you reblock a dump, use the DROPDUMP subcommand to erase the dump's description. IPCS will reinitialize the dump the first time you enter a subcommand to process the dump.

You can copy a dump using the COPYDUMP subcommand. To prevent reblocking, create and allocate the output data set before performing the COPYDUMP. Otherwise, COPYDUMP creates a dump data set with the name specified on OUTDATASET and the defaults RECFM=F and BLKSIZE=4160.

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 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.

Index

Numerics

3800 print font for print or TOC data set [131](#)
64-bit dump
 example [56](#)

A

access IPCS
 in a batch job [125](#)
access method services
 data set directory [146](#)
 problem directory [143](#)
 REPRO command
 data set directory [147](#)
 problem directory [144](#)
accessibility
 contact IBM [151](#)
active storage
 IPCS processing [6](#)
 storage map [94](#)
address
 add using a REXX function [115](#)
ADDRESS IPCS instruction [109](#)
address of data in storage [99](#)
address space
 specify on entry panel of browse option [47](#)
ALCWAIT IPCS verb name [28](#)
allocate
 IPCS print output data set [129](#)
 IPCS table of contents output data set [129](#)
ALLOCATE command
 for a print data set [129](#)
 for a table of contents data set [129](#)
ALLOCATE TSO/E command
 for a dump directory [86](#)
ALTLIB command
 invoke from IPCS dialog [104](#)
analysis
 of dump through IPCS dialog [26](#)
ANALYZE IPCS subcommand [27](#)
AOMDATA IPCS verb name [28](#)
APPCDATA IPCS subcommand [28](#)
ASCHDATA IPCS subcommand [28](#)
ASMCKECK IPCS subcommand [28](#)
ASMDATA IPCS verb name [28](#)
assistive technologies [151](#)
attention processing in IPCS
 for REXX EXECs [114](#)
AVMDATA IPCS verb name [29](#)

B

batch LSR subsystem
 use with IPCS dump directory [86](#)
batch mode processing [125](#)
batch processing

batch processing (*continued*)
 submit jobs from IPCS dialog [40](#)
BDAM (basic direct access method) [149](#)
block size
 required for IPCS processing [5](#)
BLS18099D message [18](#)
BLS18160D message [18](#)
BLSCBSAA CLIST [107](#)
BLSCBSAP CLIST [107](#)
BLSCBSVA CLIST [107](#)
BLSCBSVB CLIST [106](#)
BLSCBSVP CLIST [107](#)
BLSCBSYA CLIST [107](#)
BLSCBSYB CLIST [106](#)
BLSCBSYP CLIST [107](#)
BLSCCOMP CLIST [108](#)
BLSCDDIR CLIST
 changing current dump directory [96](#)
 changing dump directory attributes [89](#)
 using to create a dump directory [84](#)
BLSCECT parmlib member [8](#)
BLSCEDUM CLIST [108](#)
BLSCMAP CLIST [108](#)
BLSCEPTR CLIST
 invoking from the IPCS dialog [104](#)
BLSCESYM CLIST [108](#)
BLSCPCSA CLIST [108](#)
BLSCPNUC CLIST [108](#)
BLSCPRIV CLIST [108](#)
BLSCPRNT CLIST [107](#)
BLSCPSQA CLIST [108](#)
BLSCRNC2 CLIST [108](#)
BLSCRNCH CLIST [108](#)
BLSCSCAN CLIST [40](#), [106](#)
BLSCSETD CLIST [108](#)
BLSGSCMD dialog program
 used in IPCS CLISTs [119](#)
BLSXWHER exec [108](#)
BPAM (basic partitioned access method) [149](#)
browse
 of IPCS source [26](#)
BSAM (basic sequential access method) [149](#)

C

central storage
 IPCS processing [6](#)
CHARS(DUMP) printer font [129](#), [131](#)
CLEAR option
 of COPYDUMP subcommand [127](#), [137](#)
CLIST (command list)
 analyze dumps [105](#)
 controlling output [120](#)
 creating a dump directory [89](#)
 invoke from IPCS dialog [41](#)
 invoke from IPCS session [103](#)
 print detailed dump reports [107](#)

- CLIST (command list) *(continued)*
 - print dump storage [107, 108](#)
 - print screening dump reports [107](#)
 - process control blocks [108](#)
 - requesting ISPF dialog services [119](#)
 - retrieve data from a dump [108](#)
 - run a save area chain [107](#)
 - writing for IPCS [119](#)
- CLOSE subcommand
 - print output data set [131](#)
- COMCHECK IPCS subcommand [29](#)
- common storage
 - IPCS processing [6](#)
- COMPARE subcommand [108](#)
- component analysis
 - in IPCS dialog [27](#)
 - IPCS dialog option
 - list of options [28](#)
 - online help [60](#)
- component trace
 - IPCS data set processing [18](#)
- compression
 - use for dumps [6](#)
- contact
 - z/OS [151](#)
- control block
 - validate [95](#)
- CONTROL NOMSG statement
 - used in IPCS CLISTs [120](#)
- copy
 - dump data set [31](#)
 - dump directory data [31](#)
 - dumps on tape in batch [127](#)
 - GTF trace data set [31](#)
- COPYDDIR IPCS subcommand [31](#)
- COPYDDIR subcommand
 - example used in a REXX EXEC [114](#)
- COPYDUMP IPCS subcommand [31](#)
- COPYDUMP subcommand
 - example in batch [127](#)
 - example of CLEAR option in batch [127](#)
- COPYTRC IPCS subcommand [31](#)
- COPYTRC subcommand
 - example in batch [128](#)
- COUPLE IPCS subcommand [29](#)
- CPUTRACE IPCS subcommand [30](#)
- CTRACE IPCS subcommand [29, 30](#)

D

- DAE data set
 - manage [33](#)
- DAEDATA IPCS verb name [29](#)
- DASD (direct access storage device)
 - dump processing [149](#)
 - performance in dump processing [149](#)
- data description parameter
 - description [99](#)
 - parameter used in syntax diagrams [101](#)
 - resolve defaults [101](#)
 - types of parameters [99](#)
- Data Privacy for Diagnostics
 - description [62](#)
 - how to use [62](#)

- data set
 - characteristics for IPCS dump processing [5](#)
 - characteristics for IPCS trace processing [6, 18](#)
 - data set directory [146](#)
 - list for IPCS [32](#)
 - management [146, 149](#)
 - problem directory [143](#)
 - scratching [147](#)
 - types available for IPCS processing [149](#)
- data stack
 - adding IPCS subcommands [105](#)
- data-descr parameter [101](#)
- default
 - in IPCS processing [15](#)
 - printer font [131](#)
 - PROFILE-defined values [133](#)
 - setting values
 - from IPCS dialog [24](#)
 - from IPCS line mode [122](#)
 - overview [15](#)
- DEFINE command
 - of access method services
 - to create an IPCS dump directory [85](#)
- dialog
 - how to use [23](#)
- DIVDATA IPCS subcommand [29](#)
- DLF (data lookaside facility)
 - component trace [29](#)
- DLFDATA IPCS subcommand [29](#)
- DROPMAP IPCS subcommand
 - example [95](#)
- DROPSYM IPCS subcommand
 - example [93](#)
- DSNWDMP IPCS verb exit [29](#)
- dump
 - clear [137](#)
 - copy [31, 137](#)
 - copy data from tape to DASD [40](#)
 - management [38](#)
 - special purpose data set processing [5](#)
 - validate data structures [92](#)
- dump analysis
 - saved results of processing [7](#)
 - using CLISTs and REXX EXECs [105](#)
- dump data set
 - clearing
 - description [127](#)
 - example [137](#)
 - copying
 - example [137](#)
- dump directory
 - allocate [86](#)
 - batch and interactive processing [87](#)
 - batch LSR restriction [86](#)
 - changing current dump directory [96](#)
 - changing record size [89](#)
 - changing volume serial [89](#)
 - copy [31](#)
 - create [85](#)
 - initialize using IPCSDDIR TSO/E command [86](#)
 - managing via IPCS dialog [39](#)
 - subcommands used for maintenance [87](#)
 - using [83](#)
- dump display reporter panel

- dump display reporter panel (*continued*)
 - in IPCS dialog [44](#)
 - online help [60](#)
- dump initialization
 - description [17](#)
 - messages issued [18](#)

E

- entry panel [46](#)
- EQUATE IPCS subcommand
 - create a user-defined symbol [91](#)
 - create storage map entries [94](#)
- EQUATE subcommand
 - example using data description parameters [100](#)
- EVALDEF subcommand
 - control output in CLISTs and REXX EXECs [120](#)
- EVALDUMP subcommand [108](#)
- EVALMAP subcommand [108](#)
- EVALSYM subcommand [108](#)
- EVALUATE subcommand
 - used in a REXX function [111](#)
- extended format data set
 - to hold large dumps [5](#)

F

- FLAG parameter [120](#)
- format data set
 - extended
 - to hold large dumps [5](#)
- formatted dump report
 - online help [60](#)

G

- global default
 - in IPCS processing [15](#)
- GRSDATA IPCS subcommand [29](#)
- GTF trace
 - IPCS data set processing [18](#)
- GTFTRACE IPCS subcommand [30](#)
- GTFTRACE subcommand
 - example in batch [128](#)

H

- HASMFMTH IPCS verb exit [29](#)
- HELP IPCS subcommand
 - in the IPCS dialog [59](#)
- hexadecimal number
 - add using a REXX function [115](#)
- HI immediate command
 - used in IPCS [114](#)
- HSM (Hierarchical Storage Manager)
 - moving dump data sets [149](#)

I

- immediate command
 - used in IPCS [114](#)
- IMSDUMP IPCS verb exit [29](#)
- indirect addressing

- indirect addressing (*continued*)
 - with symbols [102](#)
- initial dump analysis [106](#)
- inventory panel
 - command codes [39](#)
- IOSCHECK IPCS subcommand [29](#)
- IPCS
 - starting
 - with customized access [13](#)
 - without customized access [14](#)
- IPCS (interactive problem control system)
 - analyzing dumps [26](#)
 - batch mode [7](#)
 - batch processing [40](#)
 - browse dumps [26](#)
 - CLISTs [10](#)
 - component analysis
 - online help [60](#)
 - copy data [31](#)
 - default values [15](#), [24](#)
 - dump initialization [17](#)
 - formatted dump reports
 - online help [60](#)
 - full-screen mode [7](#)
 - host command environment [109](#)
 - introduction [3](#)
 - invoking ISPF [122](#)
 - line commands [10](#)
 - line mode [6](#)
 - manage DAE data set [33](#)
 - manage data set list [32](#)
 - manage dump data sets [38](#)
 - preparing to use [13](#)
 - primary commands [10](#)
 - requesting dialog services [119](#)
 - REXX execs [10](#)
 - session modes [6](#)
 - set default values [122](#), [133](#)
 - starting [13](#)
 - subcommands
 - online help [59](#)
 - sysplex dump directory [83](#)
 - trace processing [30](#)
 - TSO/E commands [10](#)
 - types of sources [4](#)
 - user dump directory [83](#)
- IPCS batch mode
 - running CLISTs [125](#)
 - running subcommands [126](#)
- IPCS command
 - control session parameters [121](#)
 - invoke from a CLIST [105](#)
 - invoke from a REXX EXEC [105](#)
 - PARM(nn) parameter [121](#)
 - specifying parameters [121](#)
- IPCS dialog
 - advantages over line mode [7](#)
 - analysis (option 2) [26](#)
 - browse (option 1)
 - entry panel [46](#)
 - pointer panel [47](#)
 - specify a dump source [46](#)
 - component analysis (option 2.6) [27](#)
 - customize PF key definition [61](#)

- IPCS dialog (*continued*)
 - customize primary command definition [61](#)
 - DAE utility (option 3.5) [33](#)
 - data set list utility (option 3.4) [32](#)
 - defaults (option 0) [24](#)
 - description [23](#)
 - inventory (option 4) [38](#)
 - invoke under ISPF [24](#)
 - ISPF operations [60](#)
 - online help [59](#)
 - primary and line commands
 - online help [59](#)
 - primary option menu [23](#)
 - select an option [24](#)
 - starting
 - with customized access [13](#)
 - without customized access [14](#)
 - subcommand entry (option 6) [41](#)
 - submit job for batch processing (option 5) [40](#)
 - trace processing (option 2.7) [30](#)
 - tutorial (option T) [42](#)
 - utility (option 3) [31](#)
- IPCS host command environment [109](#)
- IPCSDDIR data set
 - batch LSR restriction [86](#)
- IPCSDDIR TSO/E command [86](#)
- IPCSPARM data set [8](#)
- IPCSPR00 parmlib member [8](#)
- IPCSPRnn parmlib member [8](#), [121](#)
- IPCSPRNT data set
 - choosing a default size [121](#)
- IPCSTOC data set [129](#)
- IRLM IPCS verb exit [29](#)
- IRX0920I message [114](#)
- ISPEXEC subcommand [119](#)
- ISPF (Interactive System Productivity Facility)
 - function pool dialog variables [119](#)
 - invoking IPCS dialog [24](#)
 - invoking under IPCS [122](#)
 - operations on the IPCS dialog [60](#)
 - primary and line commands [60](#)

J

- JES2
 - data from dumps [29](#)
- JES3
 - data from dumps [29](#)
- JES3 IPCS verb exit [29](#)

K

- keyboard
 - navigation [151](#)
 - PF keys [151](#)
 - shortcut keys [151](#)

L

- length of data in storage [99](#)
- linesize adjusted [133](#)
- LIST IPCS subcommand
 - example using symbols [90](#)

- LIST subcommand
 - example using data description parameters [100](#)
- LISTDUMP subcommand
 - restriction in trace processing [18](#)
- LISTEDT IPCS subcommand [29](#)
- LISTMAP IPCS subcommand
 - example [95](#)
- LISTSYM IPCS subcommand
 - example [92](#)
- LLA (library lookaside)
 - component trace [29](#)
- LLT (link list table)
 - example retrieving from a dump [113](#)
- local default
 - in IPCS processing [15](#)
- LOGDATA IPCS verb exit [29](#)
- logrec
 - request record formatting in IPCS dialog [40](#)
- LPAMAP IPCS subcommand [29](#)

M

- MACHINE parameter of DISPLAY parameter
 - summary dump data affect [18](#)
- MERGE IPCS subcommand [29](#), [30](#)
- message
 - issued during dump initialization [18](#)
- MMSDATA IPCS verb exit [29](#)
- MTRACE IPCS verb exit [29](#), [30](#)

N

- navigation
 - keyboard [151](#)
- NOSUMMARY parameter [120](#)
- NOTE subcommand
 - used in a REXX function [113](#)
- NOTERMINAL parameter [120](#)
- NUCMAP IPCS subcommand [29](#)
- NUMERIC DIGITS instruction
 - change default for dump addresses [115](#)

O

- online help
 - choosing a component analysis option [60](#)
 - description [59](#)
 - for trace processing [31](#)
 - IPCS dialog tutorial [42](#), [59](#)
 - IPCS subcommand summaries [59](#)
 - reading formatted dump reports [60](#)
- OPEN subcommand
 - CHARS(DUMP) option [129](#)
 - with CHARS(DUMP) option [131](#)
- output from IPCS
 - control in CLISTs and REXX EXECs [120](#)
 - translate characters for display [118](#)
 - using a REXX function [113](#)
- OUTTRAP function
 - used in IPCS REXX EXECs [120](#)

P

- pagesize adjusted [133](#)
- panel
 - online help in IPCS dialog [59](#)
- parmlib member
 - BLSCECT member [8](#)
 - IPCSPR00 member [8](#)
- partitioned data set
 - management [149](#)
- PF key (program function key)
 - customization for IPCS dialog [61](#)
- pointer panel
 - of browse (option 1) [47](#)
- pointer stack
 - create symbols [93](#)
 - delete and renumber symbols [93](#)
 - display source data [48](#)
 - edit pointer [48](#)
- position of data in storage [99](#)
- primary option menu
 - of IPCS dialog [23](#)
- print
 - how to [129](#)
- print data set
 - allocate [129](#)
 - attributes [129](#)
 - close [131](#)
 - control 3800 printer font for [129](#)
 - controlling 3800 printer font for [131](#)
- PRINT parameter [120](#)
- printed page report
 - setting line and page size [133](#)
- private storage
 - IPCS processing [6](#)
- problem directory [143](#)
- problem management [143](#)
- problem state program
 - unauthorized [6](#)
- PROFILE subcommand
 - set line and page size [133](#)

R

- record format
 - required for IPCS processing [5](#)
- RENUM IPCS subcommand
 - example [93](#)
- Requesting a FEEDBACK run [74](#)
- Requesting a REPORT run [74](#)
- Requesting an ANALYZE run [67](#)
- Requesting an EXTRACT run [80](#)
- Requesting an INGEST run [75](#)
- REXX debug aids
 - used in IPCS [114](#)
- REXX exec
 - add dump addresses [115](#)
 - ADDRESS IPCS instruction [109](#)
 - analyze dumps [105](#)
 - checking IPCS subcommand return codes [110](#)
 - controlling output [120](#)
 - customize control block analysis [117](#)
 - debug aids [114](#)
 - displaying output in IPCS [113](#)

REXX exec (continued)

- dump analysis techniques [115](#)
- follow a pointer chain [115](#)
- format a control block [116](#)
- function to retrieve dump data [111](#)
- invoke from IPCS dialog [41](#)
- performance consideration in IPCS [113](#)
- SIGNAL ON HALT instruction [110](#)
- suppressing unwanted error messages [110](#)
- suppressing unwanted trace records [110](#)
- TRACE OFF instruction [110](#)
- translate dump characters [118](#)
- writing for IPCS [109](#)

REXX EXEC

- invoke from IPCS session [103](#)

RSMDATA IPCS subcommand [29](#)

RUNCHAIN subcommand [108](#)

S

SADMPMSG IPCS verb exit [29](#)

save area

- format in a dump [107](#)

SCAN subcommand

- example [95](#)

scroll

- in browse option of IPCS dialog [51](#)

seed record [144](#), [147](#)

SELECT IPCS subcommand

- example [96](#)

selection code

- from storage panel in IPCS dialog [51](#)

sequential data set

- IPCS processing [6](#)
- management [149](#)

session parameter

- BLSCECT parmlib member [8](#)
- IPCSPR00 parmlib member [8](#)

SETDEF IPCS subcommand

- establish a default source [39](#)
- set defaults in IPCS dialog [25](#)

SETDEF subcommand

- control output in CLISTs and REXX EXECs [120](#)
- use in trace processing [18](#)

shortcut keys [151](#)

SMSDATA IPCS verb exit [29](#)

source

- list of types for IPCS processing [4](#)
- specify in browse (option 1) [46](#)

space requirements

- problem directory [143](#)

special purpose processing [5](#)

spin loop diagnosis

- scenario for obtaining information [135](#)

split-screen processing

- concurrent stack update [60](#)

SRMDATA IPCS verb exit [29](#)

STACK IPCS subcommand

- example [93](#)

stand-alone dump

- CLIST for initial analysis [106](#)
- copy from tape to DASD [40](#), [149](#)

STATUS IPCS subcommand [27](#)

storage

- storage (*continued*)
 - describe data using data description parameters [99](#)
- storage key
 - summary dump data affect [18](#)
- storage map
 - associated subcommands [95](#)
 - create entries [94](#)
 - create entries for an address space [96](#)
 - delete entries [95](#)
 - description [94](#)
 - entries created by IPCS [94](#)
 - list entries [95](#)
- storage panel
 - in browse option of IPCS dialog [50](#)
 - primary commands in IPCS dialog [54](#)
 - selecting words in IPCS dialog [51](#)
- STRDATA IPCS subcommand [29](#)
- striping
 - use for dumps [5](#)
- subcommand
 - entering in IPCS dialog [41](#)
- SUMDUMP IPCS verb exit [29](#)
- summary dump data
 - include in IPCS processing [18](#)
- SUMMARY IPCS subcommand [27](#)
- summary of changes [xvii](#)
- SVC dump
 - CLIST for initial analysis [106](#)
- symbol
 - naming conventions in IPCS [91](#)
- symbol table
 - create a user-defined symbol [91](#)
 - delete entries [93](#)
 - entries created by IPCS [90](#)
 - in dump directory for IPCS
 - description [90](#)
 - IPCS subcommands used for maintenance [91](#)
 - list entries [92](#)
 - naming conventions for IPCS special symbols [91](#)
 - pointer stack [93](#)
 - validate entries [92](#)
- SYMPTOM IPCS verb exit [29](#)
- SYMPTOMS IPCS verb exit [26](#)
- SYS1.DUMPnn
 - managing data sets [148](#)
- SYS1.SBLSCLI0 system library [103](#)
- SYSMDUMP dump
 - CLIST for initial analysis [106](#)
- SYSOUTTRAP command
 - used in IPCS CLISTs [119](#)
- sysplex dump directory
 - managing via IPCS dialog [39](#)
 - using [83](#)
- system dump
 - performance in IPCS data set processing [149](#)
- system dump data set processing
 - performance
 - DASD [149](#)
 - tape [149](#)
- SYSTRACE IPCS subcommand [29, 30](#)

T

table of contents data set

- table of contents data set (*continued*)
 - allocate [129](#)
 - attributes [129](#)
 - control 3800 printer font for [129](#)
 - controlling 3800 printer font for [131](#)
- tape
 - IPCS data set processing [149](#)
 - performance in dump processing [149](#)
- TCB (task control block)
 - follow a chain in dump [115](#)
- trace
 - combine data sets in batch mode [128](#)
 - copy [31](#)
- trace data set
 - improve performance during IPCS processing [18](#)
- trace data set processing [6, 18](#)
- trace processing
 - in IPCS dialog [30](#)
 - merging traces [30](#)
- trademarks [156](#)
- TSO subcommand
 - used with a CLIST [119](#)
- TSO/E (Time Sharing Option Extensions)
 - commands for IPCS [9](#)
 - environment [9](#)
- TSO/E authorized command
 - used in IPCS CLISTs [119](#)
- TSODATA IPCS verb exit [29](#)
- tutorial [59](#)

U

- user dump directory
 - managing via IPCS dialog [39](#)
 - using [83](#)
- user interface
 - ISPF [151](#)
 - TSO/E [151](#)
- utility
 - for IPCS data set list [32](#)
 - of IPCS [31](#)

V

- VERBEXIT MTRACE IPCS subcommand [30](#)
- VERBEXIT SYMPTOMS IPCS subcommand [26](#)
- VLF (virtual lookaside facility)
 - component trace [30](#)
- VLFDATA IPCS subcommand [30](#)
- VSAM (virtual storage access method)
 - IPCS data set processing [6](#)
- VSMDATA IPCS verb exit [30](#)
- VTAMMAP IPCS verb exit [30](#)

W

- WHERE subcommand [108](#)
- WLMDATA IPCS subcommand [30](#)

X

- XESDATA IPCS subcommand [30](#)



Product Number: 5655-ZOS

SA23-1384-70

