

z/OS Communications Server
2.5

IP User's Guide and Commands



Note:

Before using this information and the product it supports, be sure to read the general information under [“Notices” on page 405](#).

This edition applies to Version 2 Release 5 of z/OS® (5650-ZOS), and to subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2021-09-14

© **Copyright International Business Machines Corporation 2000, 2021.**

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

Contents

| | |
|---------------------|-----------|
| Figures..... | xi |
|---------------------|-----------|

| | |
|--------------------|-------------|
| Tables..... | xiii |
|--------------------|-------------|

| | |
|---------------------------------|-----------|
| About this document..... | xv |
|---------------------------------|-----------|

| | |
|--|------|
| Who should read this document..... | xv |
| How this document is organized..... | xv |
| How to use this document..... | xvi |
| How to contact IBM service..... | xvi |
| Conventions and terminology that are used in this information..... | xvi |
| How to read a syntax diagram..... | xvii |
| Prerequisite and related information..... | xx |

| | |
|---|------------|
| Summary of changes for IP User's Guide and Commands..... | xxv |
|---|------------|

| | |
|---|-----|
| Changes made in z/OS Communications Server Version 2 Release 5..... | xxv |
| Changes made in z/OS Communications Server Version 2 Release 4..... | xxv |
| Changes made in z/OS Communications Server Version 2 Release 3..... | xxv |

| | |
|--|----------|
| Chapter 1. Getting started with TCP/IP..... | 1 |
|--|----------|

| | |
|---|---|
| Understanding TCP/IP..... | 1 |
| Understanding TCP/IP: The physical network..... | 1 |
| Understanding TCP/IP: Protocols..... | 1 |
| Understanding TCP/IP: Network devices..... | 2 |
| Understanding TCP/IP: Addresses..... | 2 |
| How TCP/IP uses networks..... | 2 |
| How TCP/IP uses networks: Local and remote nodes..... | 2 |
| How TCP/IP uses networks: Client and server..... | 2 |
| How TCP/IP uses networks: TCP/IP addresses..... | 2 |
| How TCP/IP uses networks: Network names..... | 3 |
| How TCP/IP uses networks: Ports and port numbers..... | 3 |
| Understanding what you can do with TCP/IP..... | 3 |
| Logging on to other hosts..... | 3 |
| Transferring data sets between hosts..... | 4 |
| Sending mail..... | 4 |
| Remote command execution..... | 4 |
| Printing to or from other hosts..... | 4 |
| What you need to get started with TCP/IP..... | 5 |
| Testing commands with loopback..... | 5 |
| Obtaining command help..... | 6 |

| | |
|--|----------|
| Chapter 2. Logging on to a host using TELNET..... | 7 |
|--|----------|

| | |
|---|----|
| Using the TSO TELNET command..... | 7 |
| TELNET command | 8 |
| Using the TELNET subcommands..... | 10 |
| AO—Terminate output display | 10 |
| AYT—Query the connection | 11 |
| BRK—Send a Break or Attention keystroke to a host | 11 |
| HELP or ?—Display help information | 11 |
| IP—Interrupt the process | 12 |

| | |
|--|-----------|
| PA1—Send the PA1 keystroke to a host | 13 |
| QUIT—End the TELNET session | 13 |
| SYNCH—Clear the data path | 14 |
| ¢ and `—Send ASCII control characters to a host in line mode | 14 |
| Using the TELNET function keys..... | 15 |
| Transparent mode function keys..... | 15 |
| Line mode function keys..... | 15 |
| Suppressing carriage return and line feed..... | 15 |
| Using TELNET 3270 DBCS transform mode..... | 16 |
| Terminal and conversion type..... | 17 |
| Chapter 3. File Transfer Protocol (FTP)..... | 19 |
| Using FTP..... | 19 |
| FTP command — Entering the FTP environment | 19 |
| Logging in to FTP..... | 24 |
| Interpreting FTP client output..... | 25 |
| Allocating FTP input and output data sets..... | 27 |
| UTF-8 enabled control connection..... | 27 |
| NETRC data set..... | 28 |
| Environment variables accessed by FTP..... | 30 |
| FTP Help subcommands..... | 30 |
| Establishing and exiting a connection..... | 30 |
| Example of establishing and exiting a connection..... | 31 |
| Initial working directory considerations at the z/OS FTP server..... | 32 |
| Obtaining status and system information..... | 33 |
| Working with directories on the remote host..... | 34 |
| Examples of the CD subcommand..... | 34 |
| Examples showing the differences between DIR and LS output for z/OS UNIX directories..... | 35 |
| Examples showing the differences between DIR and LS output with DIRECTORYMode and DATASetmode for MVS..... | 36 |
| Working with directories on the local host..... | 38 |
| Security issues when using FTP..... | 39 |
| Using security mechanisms..... | 39 |
| Using a SOCKS server..... | 41 |
| FTP client security user exits..... | 41 |
| Chapter 4. Transferring data using the File Transfer Protocol (FTP)..... | 43 |
| Preparing the environment for FTP..... | 43 |
| Transferring data with FTP..... | 45 |
| How to transfer data with FTP..... | 45 |
| Examples of Get, MGet and MVGet subcommands..... | 46 |
| Examples of PUT, MPut and MVSPut subcommands..... | 52 |
| ddname support with FTP..... | 56 |
| Load module transfer with FTP..... | 59 |
| Changing local site defaults using FTP.DATA..... | 62 |
| Setting user-level options using FTPS.RC..... | 62 |
| Configuring the FTP client for SOCKS server..... | 63 |
| Sample FTP.DATA data set (FTCDATA)..... | 63 |
| FTP data conversion..... | 77 |
| Support for SBCS languages..... | 77 |
| FTP with traditional DBCS support..... | 78 |
| Support for MBCS languages..... | 81 |
| Specifying values for new data sets..... | 82 |
| Dynamic allocation of new data sets..... | 82 |
| Automatically generated SITE subcommand..... | 83 |
| Storage Management Subsystem (SMS)..... | 84 |
| Steps for using a DCBDSN model to create a new data set..... | 84 |

| | |
|---|-----|
| Statistics for PDS members..... | 85 |
| Generation data group support..... | 85 |
| GDG examples..... | 87 |
| Submitting FTP requests in batch..... | 88 |
| Submitting requests without input and output data sets..... | 91 |
| Submitting a batch job with concatenated files..... | 92 |
| FTP and data set cataloging..... | 92 |
| Using the EXEC interface..... | 92 |
| Issuing FTP subcommands from a file..... | 93 |
| Issuing FTP subcommands directly from the EXEC interface..... | 94 |
| FTP return codes..... | 95 |
| FTP standard return codes..... | 96 |
| FTP client error codes..... | 100 |
| FTP client error codes extended..... | 102 |
| FTP client error logging..... | 102 |
| Restarting a failed data transfer..... | 103 |
| Using z/OS UNIX System Services named pipes..... | 104 |
| Named pipes in the z/OS UNIX file system for the client..... | 106 |
| Steps to save a file as a named pipe in the z/OS FTP client UNIX file system | 107 |
| Steps for sending data from a named pipe in the z/OS FTP client UNIX file system..... | 108 |
| Named pipes in the server z/OS UNIX file system..... | 109 |
| Steps for storing a file as a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client..... | 111 |
| Steps for retrieving data from a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client..... | 112 |
| Interfacing with JES..... | 114 |
| Steps for submitting a job..... | 114 |
| Displaying the status of a job..... | 115 |
| Receiving spool output..... | 118 |
| Deleting a job..... | 120 |
| Steps for submitting a job and automatically receiving output..... | 120 |
| Terminating access to JES..... | 121 |
| JESINTERFACELEVEL differences..... | 121 |
| JES security..... | 123 |
| JES examples..... | 125 |
| Performing Db2 SQL queries with FTP..... | 130 |
| SQL data types supported by FTP..... | 130 |
| Creating the input data set..... | 131 |
| Setting the characteristics for the SQL query..... | 131 |
| Submitting the query..... | 133 |
| Examples of SQL query output..... | 134 |
| SUBSYS: Writing to BatchPipes..... | 135 |
| Steps for writing to BatchPipes..... | 136 |
| SUBSYS examples..... | 137 |

Chapter 5. FTP subcommands..... 139

| | |
|--|-----|
| ! subcommand—Invoke a z/OS UNIX System Services function..... | 143 |
| ACCT subcommand—Supply account information | 144 |
| APPEND subcommand—Append a local data set | 144 |
| ASCII subcommand—Change the data transfer type to ASCII | 146 |
| AUTH subcommand—Request security mechanism..... | 146 |
| BIG5 subcommand—Change the data transfer type to BIG5 | 146 |
| BINARY subcommand—Change the data transfer type to Image | 147 |
| BLOCK subcommand—Set the block data transfer mode..... | 148 |
| CCO subcommand—Turn off integrity protection..... | 148 |
| CD subcommand—Change the directory on the remote host | 149 |
| Changing the directory of a z/OS FTP server..... | 149 |

| | |
|--|-----|
| Changing the directory of a VM FTP server..... | 150 |
| Testing throughput with *DEV.NULL..... | 151 |
| CDUp subcommand—Change to the parent of the working directory..... | 151 |
| CLear subcommand—Set the protection level for data transfers to CLEAR..... | 152 |
| CLOSE subcommand—Disconnect from a remote host | 153 |
| COMPRESS subcommand—Set the compressed data transfer mode..... | 153 |
| CProtect subcommand— Set the protection level on commands..... | 154 |
| DEBUg subcommand—Set general trace options..... | 154 |
| DELEte subcommand—Delete files | 158 |
| DELImit subcommand—Display the file name delimiter | 158 |
| DIr subcommand—Obtain a list of directory entries | 159 |
| DUMP subcommand—Set extended trace options..... | 163 |
| EBcdic subcommand—Change the data transfer type to EBCDIC | 164 |
| EUckanji subcommand—Change the data transfer type to EUCKANJI..... | 165 |
| FEature subcommand—Query FTP server for features it supports..... | 166 |
| FIle subcommand—Set the file structure to File..... | 166 |
| Get subcommand—Copy files | 166 |
| GLob subcommand—Toggle expansion of metacharacters..... | 168 |
| HANGEul subcommand—Change the data transfer type to HANGEUL | 170 |
| HElp and ? subcommands—Display help information..... | 171 |
| Ibmkanji subcommand—Change the data transfer type to IBMKANJI..... | 171 |
| JIS78kj subcommand—Change the data transfer type to JIS78KJ | 172 |
| JIS83kj subcommand—Change the data transfer type to JIS83KJ | 173 |
| Ksc5601 subcommand—Change the data transfer type to KSC-5601 | 173 |
| LANGuage subcommand—Set the language used for FTP replies from the server..... | 174 |
| LCd subcommand—Change the local working directory | 175 |
| Testing throughput with *DEV.NULL..... | 176 |
| LMkdir subcommand—Create a directory on the local host | 177 |
| LOCSite subcommand—Specify site information to the local host..... | 179 |
| LOCStat subcommand—Display local status information | 201 |
| LPwd subcommand—Display the current working-level qualifier | 208 |
| LS subcommand—Obtain a list of file names | 208 |
| MDelete subcommand—Delete multiple files | 211 |
| MGet subcommand—Copy multiple files | 213 |
| MKDir subcommand—Create a directory on the remote host | 216 |
| MKFifo subcommand—Create a named pipe at the FTP server host..... | 219 |
| MMode subcommand—Set the data transfer mode | 220 |
| MPut subcommand—Copy multiple data sets to the remote host | 221 |
| MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes..... | 223 |
| MVSput subcommand – Copy a local data set into a remote data set name with the local data set attributes..... | 227 |
| NOp subcommand—Test the connection | 230 |
| Open subcommand—Connect to the FTP server | 230 |
| PAss subcommand—Supply a password | 231 |
| PRIVate subcommand—Set the protection level for data transfers to PRIVATE..... | 233 |
| PROMpt subcommand—Toggle interactive prompting for M* commands..... | 234 |
| PROtect subcommand—Set the protection level for data transfers..... | 234 |
| PROXY subcommand—Execute FTP subcommand on secondary control connections..... | 235 |
| PUt subcommand—Copy data sets to the remote host | 238 |
| PWd subcommand—Display the current working directory | 239 |
| QUIt subcommand—Leave the FTP environment | 240 |
| QUOTE subcommand—Send an uninterpreted string of data | 240 |
| RECOrd subcommand—Set the file structure to record..... | 241 |
| REName subcommand—Rename files | 242 |
| REStart subcommand - Restart a checkpointed data transfer..... | 242 |
| RMdir subcommand—Remove a directory on the remote host..... | 244 |
| SAfe subcommand—Set the protection level to safe..... | 244 |

| | |
|--|-----|
| SChinese subcommand—Change the data transfer type to SCHINESE..... | 244 |
| SENDPort subcommand—Toggle the sending of port information | 245 |
| SENDSite subcommand—Toggle the sending of site information | 246 |
| SItE subcommand—Send site-specific information to a host | 247 |
| SJIskanji subcommand—Change the data transfer type to SJISKANJI..... | 275 |
| SRestart subcommand—Restart a stream data transfer..... | 276 |
| STatus subcommand—Retrieve status information from a remote host..... | 278 |
| STREam subcommand—Set the stream data transfer mode..... | 285 |
| STRucture subcommand—Set the file structure | 286 |
| SUnique subcommand—Changes the storage method | 286 |
| SYstem subcommand—Display the operating system name | 287 |
| TChinese subcommand—Change the data transfer type to TCHINESE..... | 287 |
| TSO subcommand—Use TSO commands | 288 |
| TYpe subcommand—Set the data transfer type | 289 |
| UCs2 subcommand—Change data transfer type to Unicode UCS-2..... | 293 |
| User subcommand—Identify yourself to a host or change your TSO user ID password..... | 293 |
| Verbose subcommand - Toggle verbose mode..... | 296 |

Chapter 6. Sending electronic mail using the Communications Server SMTP

| | |
|--|------------|
| application..... | 297 |
| Creating mail messages on the JES spool data set..... | 297 |
| Using the SMTPNOTE command..... | 297 |
| Using the TSO TRANSMIT command to send a mail file..... | 300 |
| Using the IEBGENER utility to copy a mail file to a JES sysout file..... | 301 |
| SMTP commands..... | 302 |
| DATA command: Define the following information as data..... | 304 |
| EHLO command: Identify the domain name of the sending host to SMTP..... | 305 |
| HELO command: Identify the domain name of the sending host to SMTP..... | 305 |
| MAIL FROM command: Specify the sender of the mail..... | 306 |
| QUIT command: End SMTP processing..... | 306 |
| RCPT TO command: Specify the recipients of the mail..... | 307 |
| RSET command: Reset the SMTP processing to the initial state..... | 307 |
| STARTTLS command: Indicate the ability to negotiate the use of TLS..... | 307 |
| SMTP commands and reply codes across a TCP/IP connection..... | 308 |
| CSSMTP exit for unwanted mail..... | 308 |
| Example of receiving mail..... | 308 |
| Example of an undelivered mail notification..... | 309 |
| Example of generated error reports..... | 309 |

Chapter 7. Sending emails by using the sendmail to CSSMTP bridge.....311

Chapter 8. Using remote printing.....315

| | |
|---|-----|
| LPQ command—Request a list of the printer queue on a remote printer | 315 |
| LPR command—Print to a remote printer | 317 |
| LPRM command—Remove a job from the printer queue on a remote host | 328 |
| LPRSET command—Set the default printer and host name | 330 |
| TSO SMSG command—Monitoring the Status of LPD..... | 331 |

Chapter 9. Using GDDMXD/MVS with the X Window System..... 333

| | |
|---|-----|
| Overview of GDDMXD/MVS..... | 333 |
| GDDMXD/MVS keyboard and character set mappings..... | 333 |
| GDDM: Executable code..... | 333 |
| GDDM application limitations..... | 333 |
| GDDM display limitations..... | 334 |
| Using GDDMXD/MVS..... | 335 |
| GDDMXD command—Invoke the GDDMXD CLIST | 335 |
| Identifying the target display | 335 |

| | |
|--|------------|
| GDDMXD usage notes..... | 336 |
| Resizing the GDDMXD graphics window..... | 336 |
| GDDMXD/MVS: User-specified options..... | 336 |
| ANFontn option—Specify the X Window System font used for characters in the alphanumeric presentation space | 337 |
| CMap option—Specify whether the default color map is loaded or bypassed | 338 |
| Compr option—Control the technique used to compress bit-mapped data | 339 |
| Enter option—Override the default key mapping for Enter | 339 |
| GColornn option—Specify a color name | 339 |
| Geometry option—Specify the size and location of the initial GDDMXD graphics presentation space | 341 |
| GMCpnn option—Override GDDM multicolor patterns with workstation color names | 341 |
| HostRast option—Perform raster image processing at the System/370 host | 342 |
| NewLine option—Override the default key mapping for NewLine | 342 |
| XSync option—Request that the X Window System process one request at a time | 343 |
| ZWL option—Tell GDDMXD/MVS to draw all lines using 0-width lines | 343 |
| GDDMXD keyboard functions..... | 344 |
| GDDMXD/MVS keyboard functions..... | 344 |
| GDDMXD/MVS to X Window System keyboard functions..... | 344 |
| APL2 character set keyboard..... | 345 |
| Setting up <i>hlq.GDXAPLCS.MAP</i> | 345 |
| Chapter 10. Executing commands on a remote host..... | 347 |
| REXEC command - Execute a command on the remote host and receive the results on your local host | 347 |
| Using the NETRC data set..... | 349 |
| Submitting REXEC and RSH requests in batch..... | 350 |
| RSH command—Execute a command on a remote host and receive the results on your local host | 352 |
| RHOSTS.DATA data set..... | 354 |
| Using remote execution clients in a z/OS UNIX environment..... | 355 |
| The z/OS UNIX orexec/rexec command—Execute a command on the remote host..... | 355 |
| The z/OS UNIX orsh/rsh Command—Execute a Command on the remote host..... | 357 |
| Appendix A. Specifying data sets and files..... | 359 |
| MVS data set and file naming..... | 359 |
| Sequential data set file naming..... | 360 |
| Partitioned data set file naming..... | 360 |
| Transferring data between partitioned and sequential data sets..... | 361 |
| Data transfer methods..... | 361 |
| Transferring PDS directory information..... | 362 |
| AIX and UNIX file specifications..... | 362 |
| AS/400 operating system file specifications..... | 363 |
| VM file specifications..... | 363 |
| Appendix B. Mapping values for the APL2 character set..... | 365 |
| Appendix C. TELNET extensions..... | 371 |
| Character set cross reference table..... | 371 |
| Special key operation for TELNET..... | 373 |
| Operation of PF and PA keys with TELNET..... | 375 |
| Sense codes for special key operation with TELNET..... | 377 |
| Appendix D. Related protocol specifications..... | 381 |
| Appendix E. Accessibility..... | 401 |

| | |
|---|------------|
| Notices..... | 405 |
| Terms and conditions for product documentation..... | 406 |
| IBM Online Privacy Statement..... | 407 |
| Policy for unsupported hardware..... | 407 |
| Minimum supported hardware..... | 407 |
| Policy for unsupported hardware..... | 408 |
| Trademarks..... | 408 |
| Bibliography..... | 409 |
| Index..... | 413 |
| Communicating your comments to IBM..... | 423 |

Figures

- 1. Sample output of the MORE program—first screen.....16
- 2. Sample output of the MORE program—second screen..... 16
- 3. FTP.DATA for FTP client..... 64
- 4. JCL to run FTP in batch using data sets..... 90
- 5. Contents of an INPUT DD data set..... 91
- 6. JCL to run FTP in batch without using data sets..... 91
- 7. Job to create a new GDS in batch..... 91
- 8. Submitting an FTP batch job with concatenated input..... 92
- 9. How to issue the FTP subcommands from a data set..... 93
- 10. How to issue the FTP subcommands from a z/OS UNIX file system..... 93
- 11. How to issue FTP subcommands from an EXEC..... 94
- 12. How FTP subcommands can be solicited interactively from an EXEC..... 95

Tables

| | |
|---|-----|
| 1. TELNET subcommands..... | 10 |
| 2. ASCII control characters..... | 14 |
| 3. TELNET function keys in line mode..... | 15 |
| 4. Environment variables accessed by FTP..... | 30 |
| 5. FTP subcommands for getting help..... | 30 |
| 6. FTP subcommands for establishing and exiting a connection..... | 30 |
| 7. FTP subcommands for obtaining status and system information..... | 33 |
| 8. FTP subcommands for working with directories on the remote host..... | 34 |
| 9. FTP subcommands for working with directories on the local host..... | 38 |
| 10. Important differences between the draft, RFC, and CCCNONOTIFY levels..... | 40 |
| 11. FTP subcommands for preparing the environment..... | 43 |
| 12. FTP subcommands for transferring data..... | 45 |
| 13. Recommended methods for data transfer..... | 46 |
| 14. FTP client search orders..... | 62 |
| 15. FTP subcommands for DBCS support..... | 79 |
| 16. FTP TYPE subcommand aliases..... | 80 |
| 17. Mapping of DBCS keywords to CCSIDs..... | 81 |
| 18. FTP subcommand codes..... | 97 |
| 19. Client error codes..... | 100 |
| 20. FTP subcommands..... | 139 |
| 21. SMTP commands that are supported by CSSMTP..... | 302 |
| 22. sendmail bridge command switches..... | 311 |
| 23. Remote printing commands..... | 315 |

| | |
|--|-----|
| 24. Supported graphics presentation space sizes..... | 336 |
| 25. GDDMXD/MVS options..... | 337 |
| 26. GColors..... | 340 |
| 27. Recommended methods for data transfer..... | 362 |
| 28. Mapping values for the APL2 character set..... | 365 |
| 29. TCP/IP character set cross reference..... | 371 |
| 30. Special key conversions..... | 373 |
| 31. Sense codes..... | 377 |

About this document

This document describes how to use the applications available in z/OS Communications Server to perform the following functions:

- Log on to a remote host
- Transfer data sets
- Send and receive electronic mail
- Print on remote printers
- Display IBM® GDDM/MVS™ graphics on X Window System workstations
- Run a command on another host

The information in this document supports both IPv6 and IPv4. Unless explicitly noted, information describes IPv4 networking protocol. IPv6 support is qualified within the text.

This document refers to Communications Server data sets by their default SMP/E distribution library name. Your installation might, however, have different names for these data sets where allowed by SMP/E, your installation personnel, or administration staff. For instance, this document refers to samples in SEZAINST library as simply in SEZAINST. Your installation might choose a data set name of SYS1.SEZAINST, CS390.SEZAINST or other high-level qualifiers for the data set name.

A companion to this document is the [z/OS Communications Server: IP System Administrator's Commands](#), which contains specific system administrator commands used to monitor the network, manage resources, and maintain performance of z/OS Communications Server V2R3.

Who should read this document

This document is written for users who want to use the applications that are available in z/OS Communications Server V2R3.

Before using this document, you should be familiar with the IBM Multiple Virtual Storage (MVS) operating system, the IBM Time Sharing Option (TSO), and z/OS UNIX System Services and the z/OS UNIX shell. In addition, z/OS Communications Server V2R3 should already be installed and customized for your network. For information about installing, see the [z/OS Program Directory](#). For information about customizing, see the [z/OS Communications Server: IP Configuration Reference](#).

How this document is organized

This document contains the following:

- [Chapter 1, “Getting started with TCP/IP,” on page 1](#) contains basic information about TCP/IP and how to get started using it.
- [Chapter 2, “Logging on to a host using TELNET,” on page 7](#) describes how to use TELNET.
- [Chapter 3, “File Transfer Protocol \(FTP\),” on page 19](#) contains basic information about FTP.
- [Chapter 4, “Transferring data using the File Transfer Protocol \(FTP\),” on page 43](#) describes how to use the FTP command and its subcommands to sequentially access multiple hosts without leaving the FTP environment.
- [Chapter 5, “FTP subcommands,” on page 139](#) describes the FTP subcommands.
- [Chapter 6, “Sending electronic mail using the Communications Server SMTP application,” on page 297](#) describes how to use the mail forwarding SMTP client.
- [Chapter 8, “Using remote printing,” on page 315](#) describes the remote printing commands.
- [Chapter 9, “Using GDDMXD/MVS with the X Window System,” on page 333](#) describes GDDMXD/MVS and the GDDMXD CLIST and how to use GDDMXD/MVS user-specified options and keyboard functions.

- Chapter 10, “Executing commands on a remote host,” on page 347 describes how to use the REXEC and RSH clients.
- Appendix A, “Specifying data sets and files,” on page 359 describes the file-naming formats for the MVS, AIX®, UNIX, AS/400, and VM operating systems.
- Appendix B, “Mapping values for the APL2 character set,” on page 365 lists the GDDMXD/MVS default mapping values for the APL2® character set.
- Appendix C, “TELNET extensions,” on page 371 describes the Telnet 3270 DBCS Transform special operations.
- Appendix D, “Related protocol specifications,” on page 381 lists the related protocol specifications for TCP/IP.
- Appendix E, “Accessibility,” on page 401 describes accessibility features to help users with physical disabilities.
- “Notices” on page 405 contains notices and trademarks used in this document.
- “Bibliography” on page 409 contains descriptions of the documents in the z/OS Communications Server library.

How to use this document

To use this document, you should be familiar with z/OS TCP/IP Services and the TCP/IP suite of protocols.

How to contact IBM service

For immediate assistance, visit this website: <https://www.ibm.com/mysupport>

Most problems can be resolved at this website, where you can submit questions and problem reports electronically, and access a variety of diagnosis information.

For telephone assistance in problem diagnosis and resolution (in the United States or Puerto Rico), call the IBM Software Support Center anytime (1-800-IBM-SERV). You will receive a return call within 8 business hours (Monday – Friday, 8:00 a.m. – 5:00 p.m., local customer time).

Outside the United States or Puerto Rico, contact your local IBM representative or your authorized IBM supplier.

If you would like to provide feedback on this publication, see “[Communicating your comments to IBM](#)” on page 423.

Conventions and terminology that are used in this information

Commands in this information that can be used in both TSO and z/OS UNIX environments use the following conventions:

- When describing how to use the command in a TSO environment, the command is presented in uppercase (for example, NETSTAT).
- When describing how to use the command in a z/OS UNIX environment, the command is presented in bold lowercase (for example, **netstat**).
- When referring to the command in a general way in text, the command is presented with an initial capital letter (for example, Netstat).

All the exit routines described in this information are *installation-wide exit routines*. The installation-wide exit routines also called installation-wide exits, exit routines, and exits throughout this information.

The TPF logon manager, although included with VTAM®, is an application program; therefore, the logon manager is documented separately from VTAM.

Samples used in this information might not be updated for each release. Evaluate a sample carefully before applying it to your system.

z/OS no longer supports mounting HFS data sets (The POSIX style file system). Instead, a z/OS File System (ZFS) can be implemented. The term hierarchical file system, abbreviated as HFS, is defined as a data structure that has a hierarchical nature with directories and files. References to hierarchical file systems or HFS might still be in use in z/OS Communications Server publications.

Note: In this information, you might see the following Shared Memory Communications over Remote Direct Memory Access (SMC-R) terminology:

- RoCE Express®, which is a generic term representing IBM 10 GbE RoCE Express, IBM 10 GbE RoCE Express2, and IBM 25 GbE RoCE Express2 feature capabilities. When this term is used in this information, the processing being described applies to all of these features. If processing is applicable to only one feature, the full terminology, for instance, IBM 10 GbE RoCE Express will be used.
- RoCE Express2, which is a generic term representing an IBM RoCE Express2® feature that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing is applicable to only one link speed, the full terminology, for instance, IBM 25 GbE RoCE Express2 will be used.
- RDMA network interface card (RNIC), which is used to refer to the IBM 10 GbE RoCE Express, IBM® 10 GbE RoCE Express2, or IBM 25 GbE RoCE Express2 feature.
- Shared RoCE environment, which means that the "RoCE Express" feature can be used concurrently, or shared, by multiple operating system instances. The feature is considered to operate in a shared RoCE environment even if you use it with a single operating system instance.

Clarification of notes

Information traditionally qualified as Notes is further qualified as follows:

Attention

Indicate the possibility of damage

Guideline

Customary way to perform a procedure

Note

Supplemental detail

Rule

Something you must do; limitations on your actions

Restriction

Indicates certain conditions are not supported; limitations on a product or facility

Requirement

Dependencies, prerequisites

Result

Indicates the outcome

Tip

Offers shortcuts or alternative ways of performing an action; a hint

How to read a syntax diagram

This syntax information applies to all commands and statements that do not have their own syntax described elsewhere.

The syntax diagram shows you how to specify a command so that the operating system can correctly interpret what you type. Read the syntax diagram from left to right and from top to bottom, following the horizontal line (the main path).

Symbols and punctuation

The following symbols are used in syntax diagrams:

Symbol Description

- Marks the beginning of the command syntax.
- Indicates that the command syntax is continued.
- | Marks the beginning and end of a fragment or part of the command syntax.
- Marks the end of the command syntax.

You must include all punctuation such as colons, semicolons, commas, quotation marks, and minus signs that are shown in the syntax diagram.

Commands

Commands that can be used in both TSO and z/OS UNIX environments use the following conventions in syntax diagrams:

- When describing how to use the command in a TSO environment, the command is presented in uppercase (for example, NETSTAT).
- When describing how to use the command in a z/OS UNIX environment, the command is presented in bold lowercase (for example, netstat).

Parameters

The following types of parameters are used in syntax diagrams.

Required

Required parameters are displayed on the main path.

Optional

Optional parameters are displayed below the main path.

Default

Default parameters are displayed above the main path.

Parameters are classified as keywords or variables. For the TSO and MVS console commands, the keywords are not case sensitive. You can code them in uppercase or lowercase. If the keyword appears in the syntax diagram in both uppercase and lowercase, the uppercase portion is the abbreviation for the keyword (for example, OPERand).

For the z/OS UNIX commands, the keywords must be entered in the case indicated in the syntax diagram.

Variables are italicized, appear in lowercase letters, and represent names or values you supply. For example, a data set is a variable.

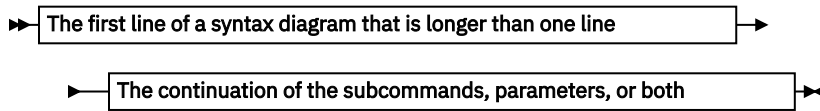
Syntax examples

In the following example, the P`U`t subcommand is a keyword. The required variable parameter is *local_file*, and the optional variable parameter is *foreign_file*. Replace the variable parameters with your own values.



Longer than one line

If a diagram is longer than one line, the first line ends with a single arrowhead and the second line begins with a single arrowhead.



Required operands

Required operands and values appear on the main path line. You must code required operands and values.

►► REQUIRED_OPERAND ◄◄

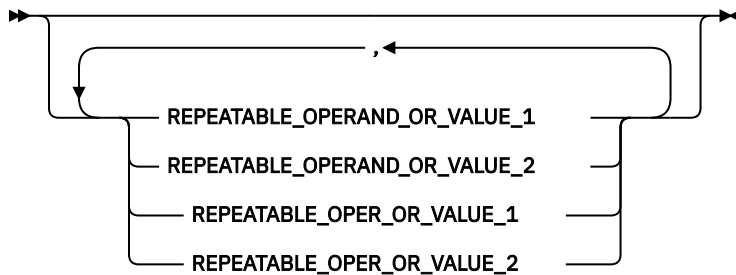
Optional values

Optional operands and values appear below the main path line. You do not have to code optional operands and values.



Selecting more than one operand

An arrow returning to the left above a group of operands or values means more than one can be selected, or a single one can be repeated.



Nonalphanumeric characters

If a diagram shows a character that is not alphanumeric (such as parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code OPERAND=(001,0.001).

►► OPERAND — = — (— 001 — , — 0.001 —) ◄◄

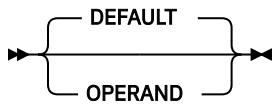
Blank spaces in syntax diagrams

If a diagram shows a blank space, you must code the blank space as part of the syntax. In this example, you must code OPERAND=(001 FIXED).

►► OPERAND — = — (— 001 — — FIXED —) ◄◄

Default operands

Default operands and values appear above the main path line. TCP/IP uses the default if you omit the operand entirely.



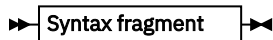
Variables

A word in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

► *variable* ◄

Syntax fragments

Some diagrams contain syntax fragments, which serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names are in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.



Syntax fragment

► 1ST_OPERAND — , — 2ND_OPERAND — , — 3RD_OPERAND ◄

Prerequisite and related information

z/OS Communications Server function is described in the z/OS Communications Server library. Descriptions of those documents are listed in “Bibliography” on page 409, in the back of this document.

Required information

Before using this product, you should be familiar with TCP/IP, VTAM, MVS, and UNIX System Services.

Softcopy information

Softcopy publications are available in the following collection.

| Titles | Description |
|-----------------------|--|
| <i>IBM Z Redbooks</i> | The IBM Z [®] subject areas range from e-business application development and enablement to hardware, networking, Linux [®] , solutions, security, parallel sysplex, and many others. For more information about the Redbooks [®] publications, see http://www.redbooks.ibm.com/ and http://www.ibm.com/systems/z/os/zos/zfavorites/ . |

Other documents

This information explains how z/OS references information in other documents.

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 \(SA23-2299\)](#). The Roadmap describes what level of documents are supplied with each release of z/OS Communications Server, and also describes each z/OS publication.

To find the complete z/OS library, visit the [z/OS library](#) in [IBM Documentation](#) (<https://www.ibm.com/docs/en/zos>).

Relevant RFCs are listed in an appendix of the IP documents. Architectural specifications for the SNA protocol are listed in an appendix of the SNA documents.

The following table lists documents that might be helpful to readers.

| Title | Number |
|--|-------------------------|
| <i>DNS and BIND</i> , Fifth Edition, O'Reilly Media, 2006 | ISBN 13: 978-0596100575 |
| <i>Routing in the Internet</i> , Second Edition, Christian Huitema (Prentice Hall 1999) | ISBN 13: 978-0130226471 |
| <i>sendmail</i> , Fourth Edition, Bryan Costales, Claus Assmann, George Jansen, and Gregory Shapiro, O'Reilly Media, 2007 | ISBN 13: 978-0596510299 |
| <i>SNA Formats</i> | GA27-3136 |
| <i>TCP/IP Illustrated, Volume 1: The Protocols</i> , W. Richard Stevens, Addison-Wesley Professional, 1994 | ISBN 13: 978-0201633467 |
| <i>TCP/IP Illustrated, Volume 2: The Implementation</i> , Gary R. Wright and W. Richard Stevens, Addison-Wesley Professional, 1995 | ISBN 13: 978-0201633542 |
| <i>TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols</i> , W. Richard Stevens, Addison-Wesley Professional, 1996 | ISBN 13: 978-0201634952 |
| <i>TCP/IP Tutorial and Technical Overview</i> | GG24-3376 |
| <i>Understanding LDAP</i> | SG24-4986 |
| z/OS Cryptographic Services System SSL Programming | SC14-7495 |
| z/OS IBM Tivoli Directory Server Administration and Use for z/OS | SC23-6788 |
| z/OS JES2 Initialization and Tuning Guide | SA32-0991 |
| z/OS Problem Management | SC23-6844 |
| z/OS MVS Diagnosis: Reference | GA32-0904 |
| z/OS MVS Diagnosis: Tools and Service Aids | GA32-0905 |
| z/OS MVS Using the Subsystem Interface | SA38-0679 |
| z/OS Program Directory | GI11-9848 |
| z/OS UNIX System Services Command Reference | SA23-2280 |
| z/OS UNIX System Services Planning | GA32-0884 |
| z/OS UNIX System Services Programming: Assembler Callable Services Reference | SA23-2281 |
| z/OS UNIX System Services User's Guide | SA23-2279 |
| z/OS XL C/C++ Runtime Library Reference | SC14-7314 |
| Open Systems Adapter-Express Customer's Guide and Reference | SA22-7935 |

Redbooks publications

The following Redbooks publications might help you as you implement z/OS Communications Server.

| Title | Number |
|--|---------------|
| <i>IBM z/OS Communications Server TCP/IP Implementation, Volume 1: Base Functions, Connectivity, and Routing</i> | SG24-8096 |

| Title | Number |
|--|---------------|
| <i>IBM z/OS Communications Server TCP/IP Implementation, Volume 2: Standard Applications</i> | SG24-8097 |
| <i>IBM z/OS Communications Server TCP/IP Implementation, Volume 3: High Availability, Scalability, and Performance</i> | SG24-8098 |
| <i>IBM z/OS Communications Server TCP/IP Implementation, Volume 4: Security and Policy-Based Networking</i> | SG24-8099 |
| <i>IBM Communication Controller Migration Guide</i> | SG24-6298 |
| <i>IP Network Design Guide</i> | SG24-2580 |
| <i>Managing OS/390 TCP/IP with SNMP</i> | SG24-5866 |
| <i>Migrating Subarea Networks to an IP Infrastructure Using Enterprise Extender</i> | SG24-5957 |
| <i>SecureWay Communications Server for OS/390 V2R8 TCP/IP: Guide to Enhancements</i> | SG24-5631 |
| <i>SNA and TCP/IP Integration</i> | SG24-5291 |
| <i>TCP/IP in a Sysplex</i> | SG24-5235 |
| <i>TCP/IP Tutorial and Technical Overview</i> | GG24-3376 |
| <i>Threadsafe Considerations for CICS</i> | SG24-6351 |

Where to find related information on the Internet

z/OS

This site provides information about z/OS Communications Server release availability, migration information, downloads, and links to information about z/OS technology

<http://www.ibm.com/systems/z/os/zos/>

z/OS Internet Library

Use this site to view and download z/OS Communications Server documentation

<http://www.ibm.com/systems/z/os/zos/library/bkserv/>

z/OS Communications Server product

The page contains z/OS Communications Server product introduction

<https://www.ibm.com/products/zos-communications-server>

IBM Communications Server product support

Use this site to submit and track problems and search the z/OS Communications Server knowledge base for Technotes, FAQs, white papers, and other z/OS Communications Server information

<https://www.ibm.com/mysupport>

IBM Communications Server performance information

This site contains links to the most recent Communications Server performance reports

<http://www.ibm.com/support/docview.wss?uid=swg27005524>

IBM Systems Center publications

Use this site to view and order Redbooks publications, Redpapers, and Technotes

<http://www.redbooks.ibm.com/>

z/OS Support Community

Search the z/OS Support Community Library for Techdocs (including Flashes, presentations, Technotes, FAQs, white papers, Customer Support Plans, and Skills Transfer information)

[z/OS Support Community](#)

Tivoli® NetView® for z/OS

Use this site to view and download product documentation about Tivoli NetView for z/OS

<http://www.ibm.com/support/knowledgecenter/SSZJDU/welcome>

RFCs

Search for and view Request for Comments documents in this section of the Internet Engineering Task Force website, with links to the RFC repository and the IETF Working Groups web page

<http://www.ietf.org/rfc.html>

Internet drafts

View Internet-Drafts, which are working documents of the Internet Engineering Task Force (IETF) and other groups, in this section of the Internet Engineering Task Force website

<http://www.ietf.org/ID.html>

Information about web addresses can also be found in information APAR II11334.

Note: Any pointers in this publication to websites are provided for convenience only and do not serve as an endorsement of these websites.

DNS websites

For more information about DNS, see the following USENET news groups and mailing addresses:

USENET news groups

comp.protocols.dns.bind

BIND mailing lists

<https://lists.isc.org/mailman/listinfo>

BIND Users

- Subscribe by sending mail to bind-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind-users@isc.org.

BIND 9 Users (This list might not be maintained indefinitely.)

- Subscribe by sending mail to bind9-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind9-users@isc.org.

The z/OS Basic Skills Information Center

The z/OS Basic Skills Information Center is a web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. The Information Center is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS systems programmer.

Specifically, the z/OS Basic Skills Information Center is intended to achieve the following objectives:

- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS

To access the z/OS Basic Skills Information Center, open your web browser to the following website, which is available to all users (no login required): <https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zbasics/homepage.html?cp=zosbasics>

Summary of changes for IP User's Guide and Commands

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Changes made in z/OS Communications Server Version 2 Release 5

This information contains no technical changes for this release.

Changes made in z/OS Communications Server Version 2 Release 4

This information contains maintenance and editorial changes for this release.

Changes made in z/OS Communications Server Version 2 Release 3

This document contains information previously presented in z/OS Communications Server: IP User's Guide and Commands, which supported z/OS Version 2 Release 2.

January 2018

Maintenance and terminology changes are made for z/OS Version 2 Release 3 in January 2018.

September 2017

New information

- sendmail to CSSMTP bridge, see [Chapter 7, “Sending emails by using the sendmail to CSSMTP bridge,” on page 311.](#)

Changed information

- sendmail to CSSMTP bridge, see [“Sending mail” on page 4.](#)

Deleted information

- Removal of SMTPD & sendmail, all references to SMTP and sendmail are deleted.

Chapter 1. Getting started with TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is a set of industry-standard protocols and applications that enable you to share data and computing resources with other computers, both IBM and non-IBM. By using TCP/IP commands at your workstation, you can perform tasks and communicate easily with a variety of other systems and workstations. z/OS Communications Server enables the user to interactively run TCP/IP applications (TCP/IP commands) from both the Time Sharing Option (TSO) and the z/OS shell.

The following subjects are covered in this topic:

- [“Understanding TCP/IP” on page 1](#)
- [“How TCP/IP uses networks” on page 2](#)
- [“Understanding what you can do with TCP/IP” on page 3](#)
- [“What you need to get started with TCP/IP” on page 5](#)
- [“Testing commands with loopback” on page 5](#)
- [“Obtaining command help” on page 6](#)

Understanding TCP/IP

TCP/IP is a set of protocols and applications that enable you to perform certain computer functions in a similar manner independent of the types of computers or networks being used. When you use TCP/IP, you are using a network of computers to communicate with other users, share data with each other, and share the processing resources of the computers connected to the TCP/IP network.

A computer network is a group of computer nodes electronically connected by some communication medium. Each node has the hardware and the programs necessary to communicate with other computer nodes across this communication medium. The node can be a PC, workstation, departmental computer, or large computer system. The size of the computer is not important. The ability to communicate with other nodes is important.

Computer networks enable you to share the data and computing resources of many computers. Applications, such as departmental file servers, rely on networking as a way to share data and programs.

Many forms of communication media are available today. Each is designed to take advantage of the environment in which it operates. Communication media consist of a combination of the physical network used to connect the computer nodes and the language, or protocol, they use to communicate with each other.

Understanding TCP/IP: The physical network

A physical network consists of electrical wiring and components, such as modems, bridges, controllers, access units, telephone lines, fiber optic cables, and coaxial cables. These are used to connect the computer nodes. The physical network can connect two nodes in a single room or thousands of nodes communicating across large geographic areas. The most common networks in use today are Local Area Networks (LANs) and Wide Area Networks (WANs). LANs cover a limited distance, generally one or two floors or buildings, while WANs, using telecommunication facilities, are used for longer distances.

Understanding TCP/IP: Protocols

Network protocols are the rules that define how information is delivered between nodes. They describe the sequence and contents of the data exchanged between nodes on the network. Network protocols determine how a computer node functions during communication with another node, how data is encoded to reach its destination safely, and what path it should follow. Protocols coordinate the flow of messages and can specify which node a message is destined for in the network. A variety of protocols are

used to take advantage of the characteristics of each of the physical network types. The most common protocols are Ethernet, 802.3, token ring, X.25, and System Network Architecture (SNA).

Understanding TCP/IP: Network devices

See the [z/OS Communications Server: IP Configuration Reference](#) for more information about network devices.

Understanding TCP/IP: Addresses

A network address is a component of the communication network and is associated with both hardware and software. The address is the means by which the sending node selects the receiving node for data transfer. It is also used by the receiving node to recognize what data is destined for it. An address is a unique code assigned to every node on a network. But an address is formed differently for different protocols. The length, position, and method used to specify an address are unique for each protocol. A communication node using one protocol cannot recognize the address of another protocol.

How TCP/IP uses networks

TCP/IP consists of a layered structure of protocols ranging from hardware-dependent programs to high-level applications. Each TCP/IP layer provides services to the layer above it and uses the services of the layer below it. The lowest layer, which is next to the physical layer, is not part of TCP/IP. This layer consists of existing protocols, such as Ethernet and token ring. TCP/IP uses the services of this layer to transport data across dissimilar networks, much like a gateway.

How TCP/IP uses networks: Local and remote nodes

A physical network is used by the hosts that reside on that network. Each host is a node on that network. A node is an addressable location in a communication network that provides host processing services. The intercommunication of these nodes gives rise to the concept of local and remote nodes. A local node pertains to a device, file, or host accessed directly from your system. A remote node pertains to a device, file, or host accessed by your system through the network.

How TCP/IP uses networks: Client and server

A server is a computer or a program that contains data or provides services to be used by other computers on the network. Some of the common server types are file, print, and mail servers. They enable your computer to share the data, devices and resources of another computer. There are also servers that provide services to let you execute programs on a computer other than your own. This enables your computer to share the processing power of another computer. Servers are also known as daemons. Generally, a server runs continuously and can handle the requests of multiple clients simultaneously.

A client is a computer or a program that requests services or data from a server. A client could, for example, request that a file located at the server be sent across the network to the client.

How TCP/IP uses networks: TCP/IP addresses

An address enables data to be routed to the chosen destination. Each destination in your network, as well as any other TCP/IP network you have access to, can be uniquely identified by its assigned address (either a 32-bit IPv4 address in dotted decimal notation, or a 128-bit IPv6 address in colon hexadecimal notation).

- IPv4 TCP/IP address

An IPv4 TCP/IP address is a 32-bit number written in dotted decimal notation. This scheme is numeric and consists of four groups separated by a period (.). For example, 9.67.1.100 represents a single host on a single network. 193.5.86.9 represents another host on another network.

- IPv6 TCP/IP address

An IPv6 TCP/IP address is a 128-bit number written in colon hexadecimal notation. This scheme is hexadecimal and consists of eight 16-bit pieces of the address. For example, `x:x:x:x:x:x:x:x` represents a single host on a single network. Alternate notations described in RFC 2373 are acceptable. For example, `FEDC:BA98:7654:3210:FEDC:BA98:7654:321` or `::1`.

To indicate IPv6 prefixing use a slash followed by the number of prefix bits. For example, use `12AB:0:0:CD30::/60` to indicate the prefix `X'12AB00000000CD3'`.

Most TCP/IP commands require you to include the address of the remote host where the server you want to access resides. Each link (physical or virtual) on a host has an IP address.

How TCP/IP uses networks: Network names

An alternative to supplying a numeric address is to use the host name, rather than the address, in TCP/IP commands. (Each host may be assigned at least one name.) Your local host can resolve the name you supply in a command into the correct numeric address. The names are translated using either a translation file or an application known as a name server. Your ability to use network names depends on how your TCP/IP network has been designed and which features have been installed.

How TCP/IP uses networks: Ports and port numbers

The use of ports and their identifying numbers are an extension to the addressing scheme. Once the address is used to deliver data to the wanted host on the network, the port number is used to identify the process for which the data is used. This enables one host to provide more than one service.

How you define the port number depends on your configuration. Some applications make use of standard, or well-known, port numbers. Two applications at the same address cannot use the same port number. If you are configuring your system with multiple instances of TCP/IP on the same system, however, they will have different addresses and therefore the same port number can be used for the same function on each stack.

TCP/IP assumes the well-known port number unless you explicitly specify otherwise when entering a TCP/IP command. A port number is entered as a decimal number on TCP/IP commands. For those cases when you are requesting the services of a user-developed server, you need to know the port number of that server.

Understanding what you can do with TCP/IP

You can perform many functions from either TCP/IP environment: TSO or z/OS UNIX System Services (z/OS UNIX). See "How to read a syntax diagram" for additional information about command syntax.

z/OS Communications Server commands provide a set of basic functions that include:

- Logging on to other hosts
- Transferring data sets and files between hosts
- Sending and receiving mail
- Using other hosts
- Printing to or from other hosts

Logging on to other hosts

The Telnet protocol provides a standardized interface that enables terminal devices and terminal-oriented processes on hosts that support TCP/IP to communicate with each other. The TSO TELNET command runs the MVS Telnet client that enables you to log on to a remote host from TSO as though you are directly attached to that host. This client supports the Telnet 3270 protocol. The MVS Telnet client does not run in the z/OS UNIX environment.

Connecting to the z/OS UNIX Telnet server from any client results in a session with the z/OS shell as if the user had entered UNIX System Services from TSO in line mode or character mode. Once the z/OS

UNIX Telnet session has been established, you can enter any UNIX System Services command that can be issued from within the z/OS shell.

See [Chapter 2, “Logging on to a host using TELNET,” on page 7](#) for more information about TELNET.

Transferring data sets between hosts

When data is created or stored at one host but is processed by another host, some method for transferring the data between hosts is necessary. TCP/IP implements File Transfer Protocol (FTP), a protocol for transferring files between any two hosts. The FTP command invokes the z/OS FTP client. From the FTP client, you can copy data sets and files between your local host and any host that implements an FTP server.

The FTP command provides subcommands that enable you to change the local and remote directories, set the transmission character code, list remote files, delete remote files, and send and receive files between hosts. You can use z/OS FTP client to perform Structured Query Language (SQL) queries and to submit jobs to JES for batch processing, as well as file transfer. FTP includes optional security features such as Kerberos and TLS authentication, and server user ID and password verification.

See [Chapter 3, “File Transfer Protocol \(FTP\),” on page 19](#), for a complete list of FTP functions.

Sending mail

The Simple Mail Transfer Protocol (SMTP) is a TCP/IP application that is used to transport electronic mail. Electronic mail enables you to send notes, messages, letters, or correspondence to others on the network. It is similar to sending a letter through the post office. You compose the message just as you would an ordinary letter, address the letter to one or more people and possibly carbon copy others. You enclose copies of the letter in envelopes, address them to the recipients, and give them to the delivery system. You expect the mail to be delivered to the correct address available for pickup when the recipient is ready. And you want any undeliverable mail returned to you. You can even keep a log of the mail you send. The following commands are available to let you send mail:

| Command | Description |
|-----------------|--|
| SMTPNOTE | Composes and sends mail from you to users on local or remote hosts. The TSO SMTPNOTE command helps you address the mail, set up a copy list, and enter the text of the message. The date, time, and your address are included automatically. |

See [Chapter 6, “Sending electronic mail using the Communications Server SMTP application,” on page 297](#) for more information about the mail-forwarding SMTP client.

Remote command execution

Just as there are occasions when you want to transfer data to a host where it can be processed, there are also occasions when you want to process the data where it exists and send the processing results to another host. The data sets or files could be too large to transfer efficiently or all the data might be kept at one host for security reasons. The computing power necessary to perform some tasks could be more than your host is capable of or the only licensed copy of a required program might reside at some other host. TCP/IP provides a command that enables you to use the processing resources of other hosts. The TSO commands REXEC and RSH and the z/OS UNIX shell command `orexec` run client programs that enable you to send any command that is valid in the remote host shell environment and receive the results at the local host. The remote host must be running a rexec server, an rsh server, or both. A user ID and password provide security checking at the remote host. The command sent to the remote host must not require user interaction to complete. See [z/OS Communications Server: IP System Administrator's Commands](#) for more information about these commands.

Printing to or from other hosts

You can print reports, documents, listings, and so on completely independent of where the job or process that created them was executed by routing the data sets to a remote host for printing. Four

TSO commands are provided for remote printing. These TSO commands do not run in the z/OS shell environment and do not support printing of z/OS UNIX files.

| Command | Description |
|---------------|--|
| LPQ | Enables you to query a printer queue on a remote printer. You can query a printer queue for a specific job, a specific user ID, or all the jobs in a remote printer queue. |
| LPR | Prints a data set on a remote printer. A variety of options enables you to specify how and where the data set is printed. |
| LPRM | Removes a job from a printer queue. |
| LPRSET | Specifies remote printer and remote host names when they are not specifically included in the Line Print commands. |

What you need to get started with TCP/IP

TCP/IP is a part of your z/OS system. To use it you need a TSO user ID and password. If you are already a TSO user, you can begin using TCP/IP.

In order to use z/OS Communications Server for V2R3 applications, you must be authorized to use z/OS UNIX System Services. For information about z/OS UNIX, see the [z/OS UNIX System Services User's Guide](#).

Ensure that you have the following before proceeding:

User IDs and passwords

You should have a user ID and password for each host you intend to use that requires user authorization and authentication. This includes most hosts you use. Some hosts on a TCP/IP network use a user ID of anonymous and a password of guest to permit all interested parties access to data sets contained at that host, but that is the exception and not the rule.

Host names

TCP/IP commands require that you know the name or IP address of the remote host you want to use.

Authorizations for data and programs

Your ability to access data sets and programs on remote hosts depends on the data security system used by that host. You might require authorization by the Resource Access Control Facility (RACF®) or other security programs before you can gain access to data sets, commands, or other resources on remote hosts.

Electronic mail addresses

To send mail electronically, you need the e-mail address of the users you want to send mail to and they need to know your electronic mail address to send you mail.

Printer names

You need the printer name and name of the remote host to which it is attached to print using TCP/IP.

Testing commands with loopback

In order to test your local machine, an address is reserved that always refers to your local host rather than any other hosts on a network. For IPv4, this class A network address is 127.0.0.1. For IPv6, the reserved loopback address is ::1. You can also specify loopback as the host name. Not all commands and clients shipped with z/OS Communications Server V2R3 support IPv6 addresses or hostnames that resolve to IPv6 addresses.

You can use the loopback address with any TCP/IP command that accepts IP addresses. When you issue a command with the loopback address, the command is sent out from your local host's client and continues until it reaches the IP layer on your local host. The command is then sent on to your local host's server.

Note: Any command or data that you send using the loopback address never actually goes out on any network.

You can also use a nonloopback local IP address for testing. It can be any local IP address assigned to a device, even though the device may not be active, but cannot be a multicast or broadcast address. The nonloopback address provides a faster response but may not reach the IP layer on your host.

The loopback address is commonly used as the first step in diagnosing network problems. The information you receive indicates the state of your system and checks to ensure that the client and server code for the function you are testing is operating properly. You should see the same response as for a normal, successful command. If the client or server code is not operating properly, the same message that would be returned for an unsuccessful command is returned.

Obtaining command help

Commands typically support **HELP** or **-?** options that allow users to obtain online help. Additionally, end user commands supported in the z/OS UNIX System Services shell environment support **man** pages (manual pages). For example, typing **man ftp** displays the manual pages for the **ftp** command.

Chapter 2. Logging on to a host using TELNET

The TELNET protocol provides a standardized interface that enables terminal devices and terminal-oriented processes on hosts that support TCP/IP to communicate with each other.

The following subjects are covered in this topic:

- [“Using the TSO TELNET command” on page 7](#)
- [“Using the TELNET subcommands” on page 10](#)
- [“Using the TELNET function keys” on page 15](#)
- [“Suppressing carriage return and line feed” on page 15](#)
- [“Using TELNET 3270 DBCS transform mode” on page 16](#)

Using the TSO TELNET command

When you use the TSO TELNET command you are running a Telnet client to connect to a remote host running a Telnet server. The data that is displayed on your terminal is managed by the Telnet client, which communicates using TCP/IP with the Telnet server at the remote host. As a result, the operation of your terminal can differ from what you are used to seeing when you are directly logged on to TSO or to another MVS application. For example, the remote host might be running UNIX, VM, or another operating system that provides a Telnet server. You need to use the terminal operation procedures of the remote host operating system while you have a TELNET session with that remote host.

TELNET management of your terminal for the remote host can also cause operational differences. For example, the function keys that are described in [“Using the TELNET function keys” on page 15](#) can result in different actions.

When all of the display data does not fit on your screen, Linemode displays the HOLDING message in the lower right corner of your screen. If this message appears, press the CLEAR key to see the rest of the data.

If your TELNET session ends for any reason, the following message is displayed:

```
Session ended.  <ENTER> to return to TSO.
```

If you invoke the services of the MVS Telnet 3270 server from a Telnet client that is not Telnet 3270 capable, you cannot use applications in full-screen mode. Once in line-mode, all nested Telnet sessions continue to be line mode. If you use TELNET in line-mode to access an MVS or VM TELNET server, all subsequent nested TELNET requests are automatically connected in line-mode as a start-stop TTY terminal, and transparent (full-screen) operations are not possible.

When you return to TSO, a message explaining why the TELNET session ended is displayed. The following is an example of what is displayed when you return to TSO:

```
TELNET terminated -- Foreign host is no longer responding
```

Restrictions:

- The z/OS TELNET client does not support the Secure Sockets Layer (SSL) protocol.
- The TELNET client uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.
- The TELNET client does not use the TCPSTACKSOURCEVIPA value for its local address. It uses any applicable SOURCEVIPA or SRCIP configuration.

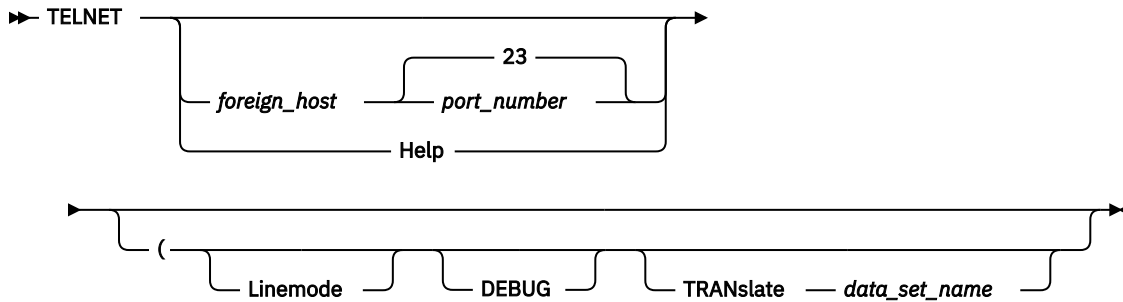
TELNET command

Purpose

The TELNET command enables you to log on to a foreign host that supports TCP/IP.

Note: For information about how to read syntax diagrams, see "How to read a syntax diagram".

Format



Parameters

foreign_host

Specifies the name or IP address of the local or remote host. If you do not specify the name or IP address of the host, you are prompted for the *foreign_host*. This must be an IPv4 address or a host name that resolves to an IPv4 address.

port_number

Specifies the port number to which you want to connect on the host. The default is well-known port 23.

Help

Provides a description of the TELNET command, its subcommands, and how it operates.

Linemode

Uses the line mode and prevents operation in the transparent mode.

In line mode, the foreign host output is displayed on your screen one line at a time, without full-screen capabilities.

Note: You cannot use the TELNET command to log on to an MVS host from an existing MVS line mode TELNET session. In this situation, the error message TELNET requires a 327x-Type terminal is displayed.

In transparent mode, the foreign host full-screen capabilities are functional on your local terminal.

Transparent mode is the default.

DEBUG

Causes TELNET client-trace data, including the data transferred to and received from the TELNET server, to be written to a data set defined by the DEBUGFIL DD statement in the user TSO LOGON procedure, or as specified by issuing the TSO ALLOC command.

The following is an example of the DEBUGFIL DD statement:

```
//DEBUGFIL DD DSN=USER28.TELNET.TRACE,DISP=OLD
```

The following is an example of the TSO ALLOC command:

```
ALLOC DDNAME(DEBUGFIL) DSNAME(USER28.TELNET.TRACE) OLD
```

TRANslate *data_set_name*

Specifies the name of a nonstandard translation table. If you specify this parameter, TELNET uses the translation table in the *user_id.data_set_name.TCPXLBIN* data set, rather than the standard translation tables *user_id.TELNET.TCPXLBIN* or *hlq.TELNET.TCPXLBIN*.

If *user_id.data_set_name.TCPXLBIN* does not exist, TELNET uses *hlq.data_set_name.TCPXLBIN*.

If *user_id.data_set_name.TCPXLBIN* and *hlq.data_set_name.TCPXLBIN* do not exist, or if they were incorrectly created, TELNET ends with an error message. A nonstandard translation table is used in line mode only.

Examples

- To log on to a host with an IP address of 1.1.2.3, enter:

```
TELNET 1.1.2.3
```

The following is displayed:

```
System:
User:    READY
System:  TELNET 1.1.2.3
        MVS TCP/IP TELNET CS V1R2
        Connecting to 1.1.2.3, port TELNET (23)
        ***
        Using Transparent Mode...

        Notes on using TELNET when in Transparent Mode:
        - To enter TELNET Command, Hit PA1
        ***
```

- If your user ID is RON and the translation table *RON.EXAMPLE.TCPXLBIN* is required rather than the standard one, you should enter:

```
TELNET 1.1.2.3 (TRANslate EXAMPLE
```

- If the remote host is neither MVS nor VM, and you specify a nonstandard translation table, a linemode connection is automatically used.
- If the remote host is an MVS or VM host and you specify a nonstandard translation table without the linemode parameter, the nonstandard translation table is ignored.
- If the host is an MVS or VM host and both the linemode parameter and a nonstandard translation table are specified, the nonstandard translation table is used.

Usage

- The minimum abbreviation for each parameter is shown in uppercase letters.
- TELNET normally operates in transparent mode. In 3270 transparent mode, all full-screen capabilities of the remote host are functional at your local display station, but the **PA1** key is the only special-function key whose intended function is passed to the application by Telnet. In line mode, the remote host output is displayed on your screen one line at a time, without full-screen capabilities.
- The TELNET command supports IBM 3270-type display stations. Examples of supported display stations are:
 - IBM 3178 Display Station
 - IBM 3179 Display Station
 - IBM 3180 Display Station
 - IBM 3191 Display Station
 - IBM 3192 Display Station

- IBM 3193 Display Station
- IBM 3194 Display Station
- IBM 3275 Display Station Model 2
- IBM 3276 Control Unit Display Station Models 2, 3, and 4
- IBM 3277 Display Station Model 2
- IBM 3278 Display Station Models 2, 3, 4, and 5
- IBM 3279 Color Display Station Models 2 and 3

Using the TELNET subcommands

You must be in the TELNET environment to use the TELNET subcommands.

To invoke a TELNET subcommand while you are logged on to the foreign host, press the PA1 key (transparent mode) or the designated PF key (line mode). After you press the PA1 or PF key, you are prompted to enter a TELNET subcommand. You can enter TELNET subcommands in uppercase or lowercase characters. [Table 1 on page 10](#) lists the TELNET subcommands.

Table 1. TELNET subcommands

| Subcommand | Description | See |
|------------|--|---|
| AO | Stops the display of information | “AO—Terminate output display ” on page 10 |
| AYt | Queries the existence of the connection | “AYT—Query the connection ” on page 11 |
| Brk | Sends a Break or Attn keystroke | “BRK—Send a Break or Attention keystroke to a host ” on page 11 |
| Help or ? | Displays help information | “HELP or ?—Display help information ” on page 11 |
| Ip | Interrupts the current process | “IP—Interrupt the process ” on page 12 |
| Pa1 | Sends a PA1 keystroke in transparent mode | “PA1—Send the PA1 keystroke to a host ” on page 13 |
| Quit | Disconnects from the foreign host | “QUIT—End the TELNET session ” on page 13 |
| Synch | Clears the data path | “SYNCH—Clear the data path ” on page 14 |

Note: The minimum abbreviation for each subcommand is shown in uppercase letters.

AO—Terminate output display

Purpose

Use the AO (Abort Output) subcommand to stop the display of output.

Format

➤ AO ➤

Parameters

There are no parameters for this subcommand.

Usage

The AO subcommand is used to clear any output that has already been produced, but has not been displayed on your terminal.

AYT—Query the connection

Purpose

Use the AYT (Are You There) subcommand to query the existence of the connection.

Format

➤ AYT ➤

Parameters

There are no parameters for this subcommand.

Usage

- You can use the AYT subcommand to check for the existence of a TELNET connection. For example, if you feel that a command is taking longer than it should to complete, issue the AYT subcommand to test whether the connection is still active.
- If the connection exists and you are operating in transparent mode, the terminal makes a sound. If you are operating in line mode, you receive a message from the TELNET server.

BRK—Send a Break or Attention keystroke to a host

Purpose

Use the BRK subcommand to send a **Break** or **Attn** (Attention) keystroke to the remote session.

Format

➤ Brk ➤

Parameters

There are no parameters for this subcommand.

Usage

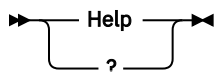
You can use the BRK subcommand to end a command without terminating the TELNET session.

HELP or ?—Display help information

Purpose

Use the HELP (or ?) subcommand to access the help facility.

Format



Parameters

There are no parameters for this subcommand.

Usage

- After your TELNET connection is established, your display station screen is controlled by the foreign host, so the help you see is different for line mode and transparent mode. The help available in transparent mode is abbreviated because TELNET does not have control of the screen.
- When you invoke the HELP or ? subcommand in line mode, TELNET displays the help information one line after another, as in the following example:

```
Once connected, follow the log in and usage
procedures of the remote host.
To invoke one of several TELNET commands, hit a
PF key (PF4-12, PF16-24), and then enter any of
the following commands:
Help or ? -- Receive (this) assistance
AYT      -- Are You There?
AO       -- Abort Output
BRK      -- Break
IP       -- Interrupt Process
SYNCH    -- Clear data path, except for TELNET commands
Quit     -- Quit the TELNET session
```

```
The following PF settings are in force:
PF1 or 13 -- Retrieve previous input line
PF2 or 14 -- Scroll halfway up
PF3 or 15 -- Turn off display of user-line; designed
              to be used before entering password
For control characters, enter %c or `c where c is:
"0": 00, "a" - "z" or "A" - "Z": 0x01-0x1A
      "2" - "6": 0x1B-0x1F
"{'": 0x5B, "}"": 0x5D, "#": 0x7F
```

- When you invoke the HELP or ? subcommand in transparent mode, TELNET overwrites one line of the current screen with the help information, as in the following example:

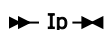
```
Valid TELNET cmds: AO,AYT,BRK,IP,PA1,QUIT,SYNCH.
```

IP—Interrupt the process

Purpose

Use the IP subcommand to interrupt the current process running on the remote host.

Format



Parameters

There are no parameters for this subcommand.

Usage

You can use the IP subcommand if you want to stop a process that is in a loop, or when you want to stop a process that you inadvertently started.

PA1—Send the PA1 keystroke to a host

Purpose

Use the PA1 subcommand to send a **PA1** keystroke to the remote session in transparent mode.

Format

➤ Pa1 ➤

Parameters

There are no parameters for this subcommand.

Usage

- The PA1 subcommand operates only in transparent mode. This subcommand replaces the **PA1** attention key on the remote host.
- When there are nested TELNET sessions, use the **PA1** key to enter a TELNET subcommand in the first open TELNET session. To enter a TELNET subcommand in the second open TELNET session, send a PA1 subcommand from the first session.
- You would normally interrupt a PING command by pressing **PA1**. However, in a transparent mode TELNET session, this key is used to invoke a TELNET subcommand. You would issue a PA1 subcommand to interrupt the PING command instead.

QUIT—End the TELNET session

Purpose

Use the QUIT subcommand to end the TELNET session.

Format

➤ Quit ➤

Parameters

There are no parameters for this subcommand.

Usage

- You should use the QUIT subcommand carefully because it can create an MVS error condition. If you do not reconnect within a timeout period, your TELNET session is canceled.
- If you are logged on to an application on a remote host, and that application is defined as disconnectable to VTAM and TCP/IP, you can use the QUIT subcommand to disconnect from the remote host without logging off the application.
- When you want to end a logon session with the host, use the logoff procedure of the host.

For more information about defining applications to VTAM and TCP/IP, see the [z/OS Communications Server: IP Configuration Reference](#).

SYNCH—Clear the data path

Purpose

Use the SYNCH subcommand to clear the data path.

Format

➤ Synch ➤

Parameters

There are no parameters for this subcommand.

Usage

The SYNCH subcommand clears the data path to the host, except for any TELNET subcommands in the data path. This subcommand enables you to ensure that commands issued when the TELNET server is inactive are not executed when the TELNET server becomes active.

¢ and `—Send ASCII control characters to a host in line mode

Purpose

Use the ¢ character and the grave accent (`) in line mode to indicate a control character.

Format

➤ ¢ *control_character* ➤
└─┬─┘
└─┬─┘
 ` `

Parameters

control_character

Indicates the ASCII control character that you want to send to the host. The purpose of each control character is specific to the remote host.

Examples

To send **Ctrl-p**, use either: ¢p or `p.

Usage

- If you want to use ¢ or ` without indicating a control character, you must enter these characters twice.
- The ASCII control characters are shown in [Table 2 on page 14](#).

Table 2. ASCII control characters

| Character input | ASCII output |
|-----------------|-------------------------------|
| `A – `Z | 01 – 1A (Ctrl-a – Ctrl-z) |
| { | 5B (left square bracket - [) |
| } | 5D (right square bracket -]) |
| `2 – `6 | 1B – 1F |

Table 2. ASCII control characters (continued)

| Character input | ASCII output |
|-----------------|--------------|
| ` # | 7F (DEL) |

Using the TELNET function keys

This section describes the functions that are assigned to PF keys when you invoke TELNET in transparent mode and line mode.

Transparent mode function keys

In transparent mode, the only function key available is the **PA1** attention key. It is used to invoke a TELNET subcommand. If there is more than one nested TELNET session, the **PA1** key is used to invoke a TELNET subcommand for the first TELNET session.

See “PA1—Send the PA1 keystroke to a host ” on page 13 for information about how to send the PA1 keystroke to the foreign host session.

Line mode function keys

Table 3 on page 15 describes the function keys that are available in line mode.

Table 3. TELNET function keys in line mode

| Function key | Description |
|--------------------------|--|
| PF4PF12, PF16PF24 | Enables you to invoke a TELNET subcommand. After pressing one of these function keys, enter a subcommand or enter Help to get a list of valid subcommands. |
| PF1, PF13 | Retrieves the previous input line, except when the line was entered in hidden mode for security reasons. |
| PF2, PF14 | Scrolls halfway up the screen. |
| PF3, PF15 | Turns off input line display so data is not echoed to the screen. For example, use either of these keys before entering your password to keep it from being displayed. |

Suppressing carriage return and line feed

It is useful if the command environment of the foreign host responds when you enter a single character, without the need for a carriage return and line feed after that character. This function is also useful when your cursor is at the end of the input field, but you want to continue the line without introducing a carriage return.

Figure 1 on page 16 and Figure 2 on page 16 show the output of a BSD UNIX program called MORE. This program displays one line or one page at a time. A carriage return character (CR) causes it to display one line, while a blank character causes it to display one page. If you are executing this program from an MVS host, use the grave accent (`) character to suppress the CR that is normally sent when you press Enter.

```
% more hosts.localNET : 4.0.0.0 : SATNET :
NET : 6.0.0.0 : YPG-NET :
NET : 7.0.0.0 : EDN-TEMP :
NET : 8.0.0.0 : BBNCCNET :
NET : 9.0.0.0 : IBM :
NET : 10.0.0.0 : ARPANET :
NET : 12.0.0.0 : ATT :
NET : 13.0.0.0 : XEROX-NET :
NET : 14.0.0.0 : PDN :
NET : 15.0.0.0 : HP-INTERNET :
NET : 18.0.0.0 : MIT-TEMP :
NET : 21.0.0.0 : DDN-RVN :
NET : 23.0.0.0 : DDN-TC-NET :
NET : 24.0.0.0 : MINET :
NET : 25.0.0.0 : RSRE-EXP :
NET : 26.0.0.0 : MILNET :
NET : 27.0.0.0 : NOSC-LCCN-TEMP :
NET : 28.0.0.0 : WIDEBAND :
NET : 29.0.0.0 : MILX25-TEMP :
NET : 30.0.0.0 : ARPAX25-TEMP :
NET : 31.0.0.0 : UCPLA-NET :
NET : 35.0.0.0 : MERIT :
--More--(0%) [HIT <ENTER> HERE. ONE LINE IS DISPLAYED]
NET : 36.0.0.0 : SU-NET-TEMP :
--More--(0%)
```

Figure 1. Sample output of the MORE program—first screen

```
NET : 39.0.0.0 : SRINET-TEMP :
--More--(0%) [HIT <BLANK>, <ACCENT GRAVE>, <ENTER>]
NET : 39.0.0.0 : SRINET-TEMP :
NET : 41.0.0.0 : BBN-TEST-A :
NET : 42.0.0.0 : CAN-INET :
NET : 44.0.0.0 : AMPRNET :
NET : 46.0.0.0 : BBNET :
NET : 128.1.0.0 : BBN-TEST-B :
NET : 128.2.0.0 : CMU-NET :
NET : 128.3.0.0 : LBL-IP-NET1 :
NET : 128.4.0.0 : DCNET :
NET : 128.5.0.0 : FORDNET :
NET : 128.6.0.0 : RUTGERS :
NET : 128.7.0.0 : KRAUTNET :
NET : 128.8.0.0 : UMDNET :
NET : 128.9.0.0 : ISI-NET :
NET : 128.10.0.0 : PURDUE-CS-EN :
NET : 128.11.0.0 : BBN-CRONUS :
NET : 128.12.0.0 : SU-NET :
NET : 128.13.0.0 : MATNET :
NET : 128.14.0.0 : BBN-SAT-TEST :
NET : 128.15.0.0 : S1NET :
NET : 128.16.0.0 : UCLNET :
NET : 128.17.0.0 : MATNET-ALT :
--More--(1%)
```

Figure 2. Sample output of the MORE program—second screen

Using TELNET 3270 DBCS transform mode

When 3270 DBCS transform mode is configured for the MVS TELNET server, all new line-mode sessions to the server are introduced with a panel where you can select transform mode or line mode.

TELNET 3270 DBCS transform mode is used to provide 3270 DBCS emulation, while the 3270 processing is done only at the host end of the connection. This enables full-screen access from non-3270 terminals.

TELNET 3270 DBCS transform mode supports terminals of the VT100/VT220 family of terminals, including VT100 and VT282.

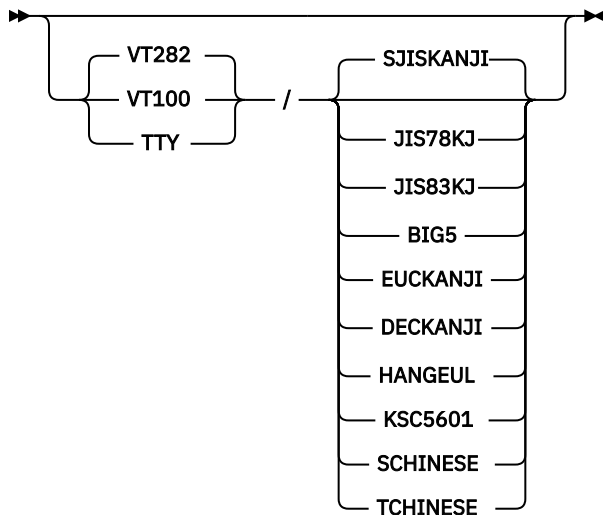
To log on to the server using 3270 DBCS transform mode, specify the LINEMODE option on the TELNET client command line. The following banner is displayed if transform mode is available:

IBM TCP/IP TELNET SERVER DBCS SERVICE START AT HH.MM.SS ON MM/DD/YY
KEY-IN YOUR TERMINAL TYPE & CONVERSION TYPE:

Terminal and conversion type

Format

When the DBCS banner appears, enter the required terminal and conversion type.



Parameters

VT100

VT100 terminal type, full-screen mode—no DBCS support

VT282

VT282 terminal type, full-screen mode—with DBCS support

TTY

Line mode—no DBCS support. This option bypasses operation in transform mode.

SJISKANJI

Shift JIS Kanji DBCS conversion

JIS78KJ

JIS Kanji 1978 DBCS conversion

JIS83KJ

JIS Kanji 1983 DBCS conversion

BIG5

Big-5 DBCS conversion

EUCKANJI

Extended UNIX code Kanji DBCS conversion

DECKANJI

DEC Kanji DBCS conversion

HANGEUL

Hangeul DBCS conversion

KSC5601

Korean Standard code KSC-5601 DBCS conversion

SCHINESE

Simplified Chinese DBCS conversion

TCHINESE

Traditional Chinese (5550) DBCS conversion

Usage

- Do not enter any spaces between the terminal type and the slash character (/), or between the slash character (/) and the conversion type. For example, to specify a VT282 terminal with Shift JIS Kanji DBCS conversion, enter the following:

```
VT282/SJISKANJI
```

- If the conversion type is not specified, it defaults to the CODEKIND specified in the TNCBCSTM configuration data set. If neither terminal type nor conversion type is specified, the terminal type defaults to VT282 and the conversion type to the CODEKIND specified in the TNDBCSTM configuration data set.
- TELNET 3270 with DBCS transform mode supports a screen size of 24 by 80. Unpredictable results may occur when using a larger screen size.
- The maximum number of concurrent TELNET 3270 DBCS Transform connections is 250.

Context

For more information about using translation tables, see the [z/OS Communications Server: IP Configuration Reference](#).

For information about character sets, see [“Character set cross reference table” on page 371](#).

For information about the TELNET extensions for terminals other than the 3270 family, see [Appendix C, “TELNET extensions,” on page 371](#).

Chapter 3. File Transfer Protocol (FTP)

The FTP command runs the FTP client program that enables you to transfer data sets and files between your local host and another host running an FTP server. Using the FTP command and its subcommands, you can sequentially access multiple hosts without leaving the FTP client.

This topic describes:

- [“Using FTP” on page 19](#)
- [“FTP Help subcommands” on page 30](#)
- [“Establishing and exiting a connection” on page 30](#)
- [“Obtaining status and system information” on page 33](#)
- [“Working with directories on the remote host” on page 34](#)
- [“Working with directories on the local host” on page 38](#)
- [“Security issues when using FTP” on page 39](#)

Using FTP

Before transferring files between your local host and a remote host, or using any other FTP functions, you must enter the FTP environment.

You can use one of the following methods to enter the FTP environment.

- Code PGM=FTP in a batch job and pass parameters using the PARM keyword. See [“Submitting FTP requests in batch” on page 88](#) for more information.
- Enter the FTP command from TSO.
- Enter the FTP command from the z/OS UNIX shell.
- Pass the FTP command parameters to the FTP Client API. See [FTP Client API](#) information in the [z/OS Communications Server: IP Programmer's Guide and Reference](#) for complete details on the FTP Client API.

FTP command — Entering the FTP environment

Purpose

Use the FTP command to enter the FTP environment. When using the FTP Client API, omit the FTP keyword.

Guidelines:

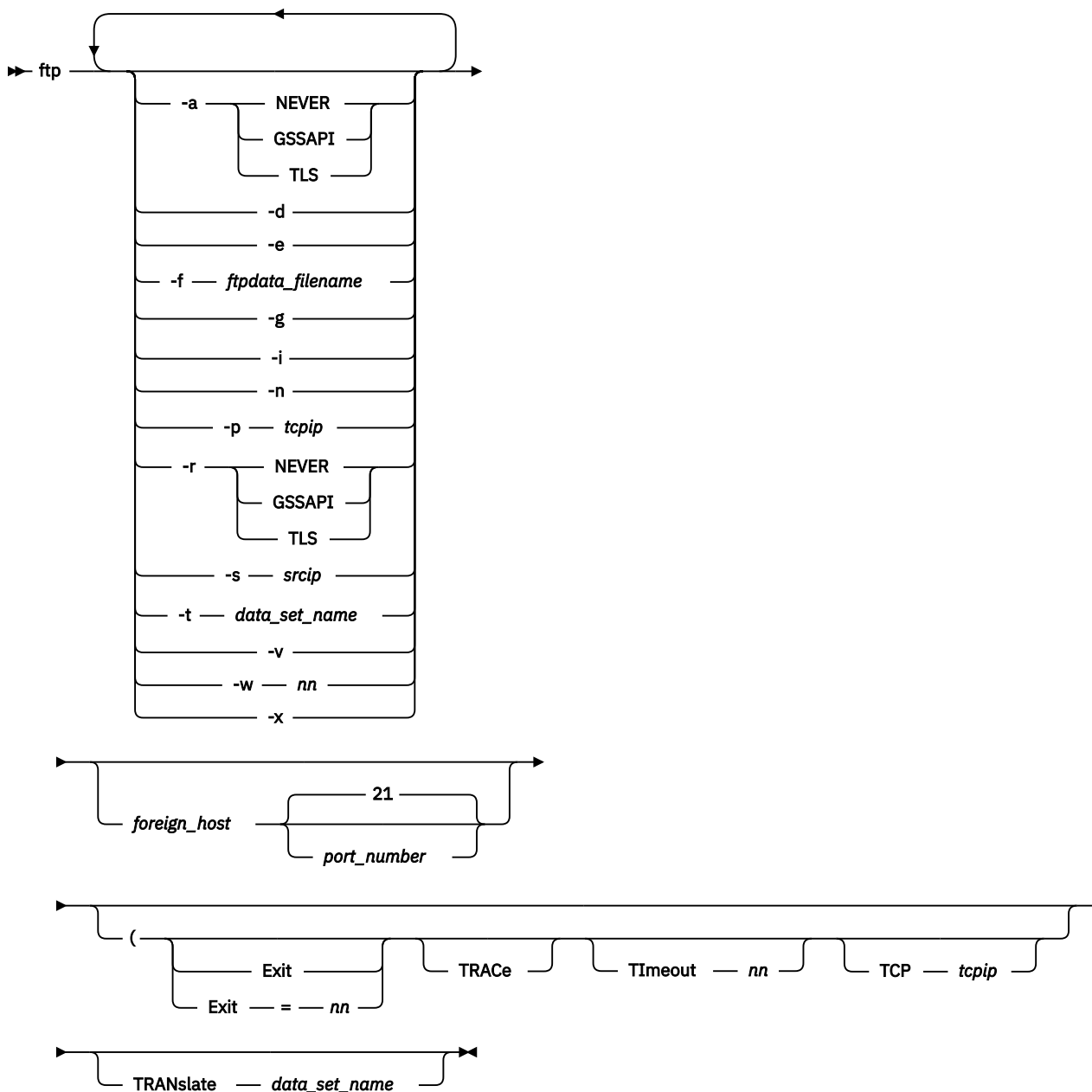
1. The FTP client expects to be invoked with POSIX(ON). If you invoke the FTP client with POSIX(OFF) you might experience unpredictable results because many of the status and result functions that are meant to inform the user of any errors during the transfer are dependent on POSIX(ON).
2. In a z/OS UNIX environment, using the FTP command in the format shown in this example results in an error:

```
ftp 1.1.2.3 (trace
```

Instead, use the standard UNIX flag (for example, -d) or precede the left parenthesis with an escape character, such as the backslash (\):

```
ftp 1.1.2.3 \ (trace
```

Format



Parameters

-a

NEVER

FTP does not attempt authentication upon initial connection.

GSSAPI

FTP attempts auto-authentication upon initial connection. FTP attempts to authenticate to the FTP server by sending the AUTH command specifying GSSAPI as the authentication type. Once the authentication type is accepted, the authentication protocol proceeds by issuing ADAT commands.

TLS

FTP attempts auto-authentication upon initial connection. FTP attempts to authenticate to the FTP server by sending the AUTH command specifying TLS as the authentication type.

Note: If you specify the TLS parameter, FTP attempts to authenticate for the control connection regardless of how you have configured FTP.DATA. FTP does not protect the data connection unless you have configured FTP.DATA to protect the data connection.

-d

Starts the generation of tracing output. Equivalent to TRACE.

-e

Terminates FTP for certain FTP errors with a nonzero MVS return code. Equivalent to EXIT.

-f *ftpdata_filename*

Specifies the client file. You can specify a z/OS UNIX file, an MVS data set, or a ddname.

Result: If you specify the -f parameter and the file or data set cannot be used, the client will exit.

Tip: The following are some examples of using the -f parameter to specify the client's FTP.DATA:

- When the FTP client is invoked from the z/OS UNIX shell:
 - To specify a z/OS UNIX file enter:
`ftp -f /etc/ftpascii myftp host`
 - To specify an MVS data set enter:
`ftp -f "'/USER12.FTP.DATA'" myftp host`
- When the FTP client is called from TSO:
 - To specify a z/OS UNIX file enter:
`ftp -f "/etc/ftpascii" 127.0.0.1`
 - To specify an MVS data set enter:
`ftp -f "'/USER1.MYFTP.DATA'" 127.0.0.1`
 - To specify an MVS PDS member enter:
`ftp -f "'/SYS1.TCPPARMS(FTPDATA)'" 127.0.0.1`
 - To specify an MVS data set by its ddname enter:
`alloc fi(myftp) da('USER1.MYFTP.DATA') SHR ftp -f "'/dd:myftp" 127.0.0.1`

Rule: When using the -f parameter from the TSO client, enclose the ftpdata parameter in quotes. For example:

- `ftp -f "/u/user1/my.ftp.data" myHost`
- `ftp -f "'/dd:ftpdd" myHost`
- `ftp -f "'/SYS1.TCPPARMS(MYFTPDATA)'"`

-g

Turns off metacharacter expansion (globbing). Equivalent to the GLOB subcommand.

-i

Turns off interactive prompting for MDELETE, MGET, and MPUT subcommands. Equivalent to the PROMPT subcommand.

-n

Inhibits automatic login, preventing the FTP client from prompting the user for a user ID and password or password phrase. If you specify the -n parameter and you have defined a NETRC data set, the data set is not used to log in to this session.

-p *tcpip*

Indicates the name of the TCP on the local host to which the FTP client should connect. This parameter is ignored if your system is not configured for multiple instances of TCP/IP. This is equivalent to TCP *tcpip*.

-t *data_set_name*

Specifies the name of a nonstandard translation table. Equivalent to TRANSLATE *data_set_name*.

-r

The option `-r` is the same as `-a` except that the AUTH command must be accepted by the server. If it is not, then the client ends the session.

Result: If you specify the `-r` parameter, FTP attempts to authenticate for the control connection regardless of how you have configured FTP.DATA. FTP does not protect the data connection unless you have configured FTP.DATA to protect the data connection.

NEVER

FTP does not attempt to authenticate upon initial connection. This option overrides a value in the FTP.DATA file that would cause authentication.

GSSAPI

FTP attempts auto-authentication upon initial connection. FTP attempts to authenticate to the FTP server by sending the AUTH command specifying GSSAPI as the authentication type. Once the authentication type is accepted, the authentication protocol proceeds by issuing ADAT commands. If the authentication type is not accepted, the client terminates the connection.

TLS

FTP attempts auto-authentication upon initial connection. FTP attempts to authenticate to the FTP server by sending the AUTH command specifying TLS as the authentication type.

-s *srcip*

Indicates the source IP address that the FTP client uses for connections. You must specify this as an IP address rather than a host name. The address must be a unicast address. INADDR_ANY, the IPv6 unspecified address (in6addr_any), IPv4-mapped IPv6 addresses, and multicast addresses are not supported. If the IP address specified is not a valid home address on the TCP/IP stack, the FTP client cannot connect to the FTP server.

Restriction: Scope information cannot be specified for the source IP address.

-v

Enables verbose mode. This parameter gives you extra information (such as message IDs) when running in z/OS UNIX.

Guideline: When running the FTP client from TSO, use the TSO profile options MSGID and NOMSGID to affix or discard message IDs.

-w *nn*

Specifies the number of seconds to be used for the TIMEOUT parameters. Equivalent to TIMEOUT *nn*.

-x

Client attempts to negotiate encryption (data and command protection level of private) immediately after a successful authentication negotiation.

foreign_host

Specifies the name of the host to which you are connecting. Specify the host by its host name or its IP address. The host can be a remote host or your local host. When you use IPv6 link-local addresses, you can provide scope information along with the host name or IP address, as described in [support for scope information](#) in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

You are prompted for a host name if you do not specify a *foreign_host* value with the FTP command. If you specify a *foreign_host* value incorrectly or if the host is not accessible, you enter the FTP environment without connecting to a host. You should then use either the OPEN subcommand to attempt another connection with a host or the QUIT subcommand (or Ctrl-C, in z/OS UNIX) to exit the FTP environment.

port_number

Specifies the port number of the FTP server on the remote host. The default is well-known port 21. The maximum port number that can be specified is 65 534. This parameter should not be used unless you are sure there is a server listening on a port other than the well-known port 21 at the destination.

Exit

Terminates FTP, for certain FTP errors, with a nonzero MVS return code. See [FTP return codes](#) for a description of the return code options available for the client.

Exit=*nn*

Terminates FTP with a nonzero return code of your choice when an FTP error occurs. Valid values are in the range 0 - 4095.

TRACe

Starts the generation of tracing output. TRACe is used in debugging.

TImeout *nn*

Specifies the number of seconds (*nn*) to be used for the following TImeout parameters:

- MYOPENTIME
- DCONNTIME
- CCONNTIME
- INACTTIME
- DATACTTIME

The name of each timer corresponds to an FTP.DATA statement available to set that timer. See the [FTP.DATA data set statements information in the z/OS Communications Server: IP Configuration Reference](#) for a description of each of these timers and its default value.

Results:

- If the value is not in the range 15 - 85600 or 0, FTP uses the default values for the TImeout parameters.
- If the value is not a number, all TImeout parameter values are set to 0.

TCP *tcpip*

Indicates the name of the TCP on the local host to which the FTP client should connect. This parameter is ignored if your system is not configured for multiple instances of TCP/IP.

Note: You must specify this value as a parameter, not as a value in the FTP.DATA data set. You can choose to specify this value with the TCPIPJOBNAME statement in the resolver configuration file.

TRANsLate *data_set_name*

Specifies the data set name of a nonstandard translation table. If you specify this parameter, FTP uses the translation table in the *user_id.data_set_name*.TCPXLBIN data set, rather than the standard translation table provided with TCP/IP (*hlq*.STANDARD.TCPXLBIN). The *hlq*.STANDARD.TCPXLBIN data set is never used if you specify the TRANSLATE parameter.

If *user_id.data_set_name*.TCPXLBIN does not exist, FTP uses *hlq.data_set_name*.TCPXLBIN. If neither data set exists, or if they were incorrectly created, FTP ends with an error message.

Since the TRANsLate parameter also dictates the search order for DBCS translation tables, you might want to use a customized DBCS translation table but not require a modified SBCS translation table. If this is the case, copy *hlq*.STANDARD.TCPXLBIN into the nonstandard TCPXLBIN translation table data set to ensure that FTP will start.

Notes:

1. Use the CTRLConn and SBDataconn statements in your local FTP.DATA data set to specify different SBCS tables for the control and data connections, or use the LOC SITE SBDataconn subcommand to change the SBCS translation for the data connection. For information on specifying these statements, see [“Support for SBCS languages” on page 77](#).
2. If you require the use of a customized DBCS translation table, but cannot or do not want to use the TRANsLate parameter, you can name the data set such that it is found in the client search order (for example, *userid*.FTP.TCPdbBIN). See [z/OS Communications Server: IP Configuration Reference](#) for information about the DBCS translation table search order. FTP does not terminate because it fails to find a nonstandard DBCS translation table data set.

Usage

- When starting FTP in a TSO environment that includes support for the REXX programming language, you receive the following message:

```
CSV003I Requested module IRXSTK not found
```

This is a normal informational message when starting FTP in a TSO environment.

- If you enter the FTP flags (-a, -d, -e, -f, -g, -i, -n, -p, -r, -s -t, -v, -w, and -x) from z/OS UNIX the flags must be entered in lowercase. These options can be entered in lowercase or uppercase from TSO.
- GSSAPI authentication is supported only for IPv4 connections. The client fails the negotiation when the connection is IPv6.
- NOOPTMSS is no longer supported and is ignored.
- When FTP is started from the FTP Client API, some of the start parameters are ignored. See the [FTP client behavior when started by the FTP Client API](#) information in the [z/OS Communications Server: IP Programmer's Guide and Reference](#) for details.

Context

- See [“Open subcommand—Connect to the FTP server”](#) on page 230 and [“QUIT subcommand—Leave the FTP environment”](#) on page 240 for more information about the OPEN and QUIT subcommands.
- See [FTP return codes](#) for a description of return code handling in the FTP client.
- See [“Changing local site defaults using FTP.DATA”](#) on page 62 for information about the FTP.DATA data set.
- See [z/OS Communications Server: IP Configuration Reference](#) for information about the TCPIP.DATA data set or loading and customizing DBCS translation tables.

Logging in to FTP

If you correctly specify a foreign host with the FTP command, you are prompted to identify yourself. The following is a sample of the information that is displayed after you successfully invoke the FTP command with *foreign_host* correctly specified.

```
IBM FTP CS V1R5
FTP: using TCP/CS
Connecting to: 9.67.113.37 port: 21.
220-FTPD1 IBM FTP CS V1R4 at vic135, 19:11:09 on 2003-01-15.
220 Connection will close if idle for more than 5 minutes.
NAME (9.67.113.37:USER10):
>>> USER USER10
331 Send password please.
PASSWORD:

>>> PASS
230 USER10 is logged on. Working directory is "/tmp".
Command
```

After successfully identifying yourself, you are prompted for a password if the foreign host requires a password. If you enter the password correctly, you are connected to the foreign host.

Tips:

- You can use the data set NETRC to automatically provide user ID, password, and accounting information while logging in to a remote host. For information about using NETRC, see [“NETRC data set”](#) on page 28.
- You can respond to the NAME prompt with the same input that the FTP client accepts as arguments of the User subcommand. See [“User subcommand—Identify yourself to a host or change your TSO user ID password”](#) on page 293 for more information about the User subcommand.

- You can respond to the PASSWORD prompt with the same input that the FTP client accepts as arguments of the PAss subcommand. See [“PAss subcommand—Supply a password”](#) on page 231 for more information about the PAss subcommand.

Results:

- If you have enabled UTF-8 encoding of the control connection, the login sequence is different. See [“UTF-8 enabled control connection”](#) on page 27 for more information.
- You might see one or more 230- replies while logging in to the FTP server. These replies contain information about the current session. Reply code 230 or 230- always indicates that you have successfully logged in to the FTP server. However, the replies 230- sometimes contain information about errors encountered while logging in. These replies indicate errors in the z/OS FTP server configuration files:

- 230- CWD cmd failed : *reason*
- 230- Unrecognized parameter *parameter* on SITE command
- 230- Unable to open FTPS.RC configuration file *error information*
- 230- Unrecognized command -*cmd*- entered
- 230- The message was truncated

Contact the system programmer for assistance. See [FTPD reply codes](#) in [z/OS Communications Server: IP and SNA Codes](#) for additional details about 230-type replies.

- When the FTP Client API is invoked by an application program, ddnames associated with the application are not available to the created FTP client process and the handling of prompts from the client differs. See [FTP client behavior when started by the FTP Client API](#) information in [z/OS Communications Server: IP Programmer's Guide and Reference](#) for details.

For the procedure to enter the FTP environment using the FTP command, see [“Establishing and exiting a connection”](#) on page 30 for more information.

Interpreting FTP client output

The z/OS FTP client output can be categorized as follows:

- Client messages
- Server replies
- Echo input
- Client trace

The following is an example of FTP client output:

```
1# ftp -v vic135
2EZY2640I Using 'USER1.FTP.DATA' for local site configuration parameters.
3EZYFT46E Error in 'USER1.FTP.DATA' file: line 580 near column 11.
4EZYFT25I Using //'TPOUSER.STANDARD.TCPXLBIN' for FTP translation tables for
    the control connection.
5EZYFT75I Using internal translate tables for the data connection.
6EZA1450I IBM FTP CS V1R7
7EZA1466I FTP: using TCPDS
8EZYFT18I Using catalog '/usr/lib/nls/msg/C/ftpdmsg.cat' for FTP messages.
9EZA1554I Connecting to: vic135.tcp.raleigh.ibm.com 9.42.103.37 port: 21.
10220-FTPD1 IBM FTP CS V1R7 at VIC135.tcp.raleigh.ibm.com, 21:21:49 on 2004-11-08.
11220 Connection will not timeout.
12EZA1459I NAME (vic135:USER1):
13EZA1701I >>> USER USER1
14331 Send password please.
15EZA1789I PASSWORD:
16230 USER1 is logged on. Working directory is "USER1.".
17EZA1460I Command:
18debug all
19PC0346 parseCmd: subcommand: debug
20PC0371 parseCmd: parameter 1: all
21EZA2923I Active client traces - FLO CMD PAR INT ACC UTL SEC FSC(1) SOC(1) SQL
22SC3392 resetLastReply: entered
23PC0657 parseCmd: using primary session.
```

```

24CU0307 getCommand: entered
25EZA1460I Command:
26debug time
27PC0346 parseCmd: subcommand: debug
28PC0693 findCmd: entered with debug
29PC0775 findCmd: command found is debug
30PC0371 parseCmd: parameter 1: time
31PC0404 parseCmd: findCmd returned the cmdrecord for debug
32PC0536 parseCmd: using primary session
33CL0205 debug: entered
34GU2981 setDebug: entered
35EZA2923I Active client traces - FLO CMD PAR INT ACC UTL SEC FSC(1) SOC(1) SQL
3616:54:46 SC3392 resetLastReply: entered
3716:54:46 PC0657 parseCmd: using primary session.
3816:54:46 CU0307 getCommand: entered
39EZA1460I Command:

```

Following are descriptions of the numbered items in the example.

1

This is the command that starts the FTP client. The -v option directs FTP to precede each client message with its unique 7-character identifier.

For more information about FTP start options, see [“FTP command — Entering the FTP environment” on page 19](#).

2 to 9

These are FTP client messages, whose identifiers are always 8 characters long. The message identifier appears only when you start FTP with the -v option. After you have started FTP, you can use the client subcommand, verbose, to toggle display of client message identifiers on and off.

The first three characters are always one of the following strings:

- EZA
- EZY
- EZZ

The final character can be E, I, or W.

See [z/OS Communications Server: IP Messages Volume 1 \(EZA\)](#), [z/OS Communications Server: IP Messages Volume 3 \(EZY\)](#), and [z/OS Communications Server: IP Messages Volume 4 \(EZZ, SNM\)](#) for more information about FTP client messages.

10, 11

These are replies from the FTP server. FTP server replies are always prefixed with a 3-digit numeric reply code, in the range 100 - 555. For the meaning of the reply code, see RFC 959. If the server is a z/OS FTP server, you will also find the FTP server reply code and reply text documented in [z/OS Communications Server: IP and SNA Codes](#).

12, 13, 15, 17, 21, 25, 39

These are FTP client messages, as indicated by the 3-letter prefix.

14, 16

These are FTP server replies, as indicated by the 3-digit prefix.

18

This is the echo of input entered in response to the FTP client command prompt at line **17**. The debug subcommand turns on the FTP client trace.

19, 20, 22 to 24

This is FTP client trace output. FTP client trace output lines are always composed of a 2-letter prefix with a 4-digit numeral. The FTP client trace is for the use of the IBM Service Center.

26

This is the echo of input entered in response to the FTP client command prompt at line **25**. The time option of the debug subcommand causes the FTP client trace to be time-stamped

27 to 34

This is client trace generated by the client processing the debug subcommand. This trace is not time-stamped because the client has not yet recognized that it is processing the TIME option.

35

This is a client message issued in response to the debug subcommand.

36 to 38

This is client trace output. At this point, the client has recognized the TIME option entered at line 26 and client trace is now time-stamped.

Allocating FTP input and output data sets

When you invoke the FTP command from TSO, a check is made to see whether a data set is allocated to INPUT. If a data set is allocated, subcommands are read from that data set rather than from your terminal. Similarly, a check is also made to see whether a data set is allocated to OUTPUT. If so, all FTP prompts and replies are written to that data set rather than to your terminal.

The record length and block size of the output data set can be any size. If the logical record length of the output data set is less than 100 bytes, some messages could be truncated or wrapped around to the next line.

If you create INPUT and OUTPUT data sets, use the following guidelines:

- Specify the INPUT data set:

- Record format=FB.
- Logical record length=2080.

The logical record length of the input data set can be any value in the range 80 - 2080.

- Block size is a multiple of logical record length.

- Specify the OUTPUT data set:

- Record format=FB.
- Logical record length=160.
- Block size is a multiple of 160.

Restriction: When the FTP Client API is invoked by an application program, INPUT and OUTPUT ddnames are not available to the created FTP client process. See the [z/OS Communications Server: IP Programmer's Guide and Reference](#) for a description of the [FTP Client API](#).

UTF-8 enabled control connection

You can specify EXTENSIONS UTF8 in the client's FTP.DATA data set to enable the FTP client to use and accept UTF-8 encoding of the control connection. See the [z/OS Communications Server: IP Configuration Reference](#) for information about the EXTENSIONS statement. If you code EXTENSIONS UTF8 in the client's FTP.DATA data set, the login sequence is different. Following is an example of logging in to a UTF-8 enabled FTP server when the client has enabled UTF-8 encoding of the control connection.

```
IBM FTP CS V1R5
FTP: using TCPDS
Connecting to: 9.67.113.37 port: 21.
220-FTPD1 IBM FTP CS V1R4 at vic135, 19:16:11 on 2003-01-15.
220 Connection will close if idle for more than 5 minutes.
>>> FEAT
211- Extensions supported
    UTF8
    LANG en*
211 End
>>> LANG en
200 - Language is en-US (United States English)
NAME (9.67.113.37:USER10):

>>> USER USER10
331 Send password please.
PASSWORD:

>>> PASS
```

```
230 USER10 is logged on. Working directory is "/tmp".  
Command:
```

The difference is that the client issues the FEAT command during login to negotiate use of UTF-8 on the control connection, as specified in RFC 2640 (see [Appendix D, “Related protocol specifications,”](#) on page 381). In this example, the FEAT reply indicates the server supports RFC 2640 (UTF8 and LANG keywords), so the client issues LANG to commence UTF-8 encoding of the control connection.

Here is an example of a UTF-8 enabled client logging in to a server which does not support UTF-8 encoding:

```
IBM FTP CS V1R5  
FTP: using TCPDS  
Connecting to: 9.67.113.37 port: 21.  
220-FTPD1 IBM FTP CS V1R4 at vic135, 19:20:43 on 2001-10-15.  
220 Connection will close if idle for more than 5 minutes.  
>>> FEAT  
211- Extensions supported  
SIZE  
MDTM  
REST STREAM  
211 End  
NAME (9.67.113.37:USER10):  
  
>>> USER USER10  
331 Send password please.  
PASSWORD:  
  
>>> PASS  
230 USER10 is logged on. Working directory is "/tmp".  
Command:
```

The client issued the FEAT command during login because EXTENSIONS UTF8 is coded in FTP.DATA. Since the server FEAT reply did not indicate the server supports RFC 2640, no LANG command was issued by the client. The client will not send UTF-8 encoded data to this server.

NETRC data set

The *user_id*.NETRC data set (*/\$HOME/.netrc*, in z/OS UNIX) provides you with an alternative to specifying your *user_id* and *password* values as FTP parameters when you want to FTP to a remote host. The following example shows you how to specify the *user_id*.NETRC data set:

```
machine mvs1.tcp.raleigh.ibm.com login user28 password user28  
machine 9.67.112.25 login user28  
machine FEDC:BA98:7654:3210:FEDC:BA98:7654:3210 login user28
```

The keywords *machine*, *login*, and *password* must be lowercase. The variables *user_ID* and *password* might be case sensitive, depending on the remote host. (For example, when using UNIX or AIX hosts, the *user_ID* and *password* values are case sensitive.) The *hostname* variable that is specified after the *machine* keyword can include scope information, as described in the [support for scope information](#) in the *z/OS Communications Server: IPv6 Network and Application Design Guide*.

Guidelines:

- If you include *scope* on the FTP command (for example, *hostname%scope*), there should be an entry defined in the *user_id*.NETRC data set that includes *scope* as part of the *hostname* value following the *machine* keyword. Defining this entry ensures that the correct *user_id* and *password* values are selected.
- Although the FTP client allows the keywords for a single machine entry to be split across multiple lines, REXEC requires all of the values to be on a single line. If you specify the ACCOUNT keyword in the NETRC file, specify the password keyword; otherwise, the FTP client uses NULL as the password and sends it to the server.

Rules:

- Code a password phrase that contains blanks in NETRC by enclosing the entire password phrase in quotation marks. You can use single or double quotation marks. If the password phrase itself contains a quotation mark, use the other style of quotation mark to enclose the password phrase.

Example: Code the password phrase *What's up, Doc?* in NETRC as "What's up, Doc?" but not as 'What's up, Doc?'.

If you code user data for the z/OS FTP server user exit FTCHKPWD in NETRC, and either user data or the password contains blanks, enclose the password and user data in quotation marks.

Example: Code the password phrase *What's up, Doc?* with the user data *FTCHKPWD exit parameter* as "What's up, Doc?:FTCHKPWD exit parameter".

- Do not use quotation marks to enclose a password phrase that is comprised only of any of the following characters:
 - Uppercase or lowercase letters
 - Numerals from 0 to 9
 - The following special characters:
 - @
 - #
 - \$
 - -
 - {
 - .
 - (
 -)
 - *
 - %
 - +

Example: Code the password phrase *JoeIBMer@ibm.com* in NETRC as *JoeIBMer@ibm.com*, but not as 'JoeIBMer@ibm.com', nor as "JoeIBMer@ibm.com".

Restriction:

- A password phrase that you code in the NETRC data set or file must not contain both single quotation mark and double quotation mark characters. You can use either style of quotation marks in the password phrase, but not both.

Example: The password phrase *What's up, Doc?* is valid because it contains only single quotation marks. Code it in NETRC as "What's up, Doc?". The password phrase "What's up, Doc?" with the double quotation marks as part of the password phrase cannot be entered at the z/OS FTP client or coded in NETRC because it contains both styles of quotation marks.

To invoke the *user_id*.NETRC data set and automatically log on to the remote host named MVS1, enter the FTP command as shown in the following example:

```
User:      ftp mvs1
System:
          IBM FTP CS V1R5
          FTP: using TCP/CS
          Connecting to: 9.67.113.61 port: 21.
          220-FTP01 IBM FTP CS V1R2 at MVS01C04, 12:00:51 on 2003-01-12.
          220 Connection will close if idle for more than 5 minutes.
          >>>USER user28
          331 Send password please.
          >>>PASS *****
          230 USER28 is logged on.  Working directory is "/u/user28".
          Command:
```

In order to bypass definitions in a *user_id*.NETRC data set for an FTP session, specify the -n parameter. For information about using the *user_id*.NETRC data set in a batch file, see [“Submitting FTP requests in batch”](#) on page 88.

Environment variables accessed by FTP

The following are environment variables that are referenced by FTP:

| Table 4. Environment variables accessed by FTP | | | |
|--|----------------------|--------------------------|---|
| | Environment variable | Command-type application | Description |
| 1 | HOME | FTP CLIENT | Initialized by the system at login to a value equal to the path name of the user's home directory. |
| 2 | LANG | FTP CLIENT | Determines the locale category for native language, local customs, and coded character set in the absence of the LC_ALL and other LC_* environment variables (including LC_COLLATE, LC_CTYPE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, AND LC_TIME). |
| 3 | NLSPATH | FTP CLIENT | Contains a sequence of templates used by the catopen() function when it attempts to locate message catalogs. Each template consists of an optional prefix, one or more conversion specifications, a file name, and an optional suffix. |
| 4 | SHELL | FTP CLIENT | Sets the default shell used by make, vi, and other tools. |

FTP Help subcommands

The FTP Help subcommands are listed in [Table 5 on page 30](#).

| Table 5. FTP subcommands for getting help | | |
|---|--|---|
| Subcommand | Description | See |
| ? | Provides an introduction to using FTP. | “HElp and ? subcommands—Display help information” on page 171 |
| HElp | Displays help information for FTP. | “HElp and ? subcommands—Display help information” on page 171 |
| man | UNIX Shell command provides help information about the z/OS UNIX FTP client. | “Obtaining command help” on page 6 |

Establishing and exiting a connection

You normally establish a connection to a foreign host when you invoke the FTP command with a *foreign_host* specified. If you are not successful in specifying a foreign host, or if you need to connect to a different foreign host, use the subcommands listed in [Table 6 on page 30](#).

| Table 6. FTP subcommands for establishing and exiting a connection | | |
|--|---|---|
| Subcommand | Description | See |
| ACct | Sends host-dependent account information. | “ACct subcommand—Supply account information ” on page 144 |

Table 6. FTP subcommands for establishing and exiting a connection (continued)

| Subcommand | Description | See |
|------------|---|---|
| CLOSE | Disconnects from the foreign host. | “CLOSE subcommand—Disconnect from a remote host ” on page 153 |
| OPEN | Opens a connection to a foreign host. | “OPEN subcommand—Connect to the FTP server ” on page 230 |
| PASS | Supplies a password or password phrase to the foreign host. | “PASS subcommand—Supply a password ” on page 231 |
| QUIT | Leaves the FTP command environment. | “QUIT subcommand—Leave the FTP environment ” on page 240 |
| USER | Identifies you to a foreign host. | “USER subcommand—Identify yourself to a host or change your TSO user ID password” on page 293 |

Example of establishing and exiting a connection

This example shows how a single FTP session can be used to connect to the following multiple foreign hosts:

1. MVS Host: 2001:0DB8:c2d4::9:67:115:12 port 21
2. VM Host: 192.9.2.4

```

ftp 2001:0DB8:c2d4::9:67:115:12
IBM FTP CS V1R5
FTP: using TCPCS
Connecting to: 2001:0DB8:c2d4::9:67:115:12 port: 21.
220-FTPD1 IBM FTP CS V1R4 at MVSVC96, 12:38:19 on 2003-01-27.
220 Connection will close if idle for more than 5 minutes.
NAME (2001:0DB8:c2d4::9:67:115:12:USER1):
user2 *****
>>> USER user2
331 Send password please.
>>> PASS
230 USER2 is logged on. Working directory is "/".
Command:
User: close
System:
>>>QUIT
221 Quit command received. Goodbye.

Command:
User: open 192.9.2.4

System:

Connecting to: 192.9.2.4 port: 21.
220-FTPSERVE IBM VM Level 320 at VM.IBM.COM, 14:03:49 EST MONDAY 2003-01-11
220 Connection will close if idle for more than 5 minutes.
NAME (<host>:tsouserid):
User: vmuser
System:
>>USER vmuser
331 Sent password please.
Password:
>>>PASS *****
230 VMUSER logged in; working directory = VMUSER 191

Command:
User: account
System:
Usage: ACCT account-information

Command:
User: acct vmuser
System:
>>>ACCT *****
230 You now have write permission to VMUSER 191

Command:
User: close
System:
>>>QUIT
221 Quit command received. Goodbye.

Command:

```

Initial working directory considerations at the z/OS FTP server

When you first log in to a z/OS FTP server, the initial or default working directory at the server is determined by the following:

- The value specified on the STARTDIRECTORY statement in the FTP.DATA file of the server
- The user ID you used when you connected to the server
- The prefix defined in the profile for the user ID

Note: To use FTP, your user ID must have an OMVS segment defined (or defaulted).

If STARTDIRECTORY HFS is defined at the server, the initial working directory is the home directory for the user ID. An example of an initial working directory for USER1 is the following:

```
/u/user1
```

If STARTDIRECTORY MVS is defined at the server and no prefix is defined for the user ID, the initial working directory is the user ID followed by a period. An example of an initial working directory for USER1 is the following:

```
USER1.
```

If your TSO user ID is defined through Resource Access Control Facility (RACF) and a PREFIX is defined for the user ID, the PREFIX value is used as the initial working directory.

The PREFIX of a TSO user ID can be set or changed by using the TSO PROFILE command as follows:

1. Log in to TSO on the MVS system of the FTP server.
2. Set your new prefix using the TSO PROFILE command:

```
TSO PROFILE Prefix(prefix)
```

where *prefix* is any TSO prefix that you choose.

Notes:

- a. You must enter both the opening and closing parentheses.
 - b. At this point, the TSO prefix is defined for your current TSO session but is not known to RACF or the FTP server until you log off and log on.
3. Log off to save the new default working directory name.

The TSO prefix should now be your default working directory whenever you log on to an FTP session on that FTP server. To verify that you set up the default working directory correctly, perform the following steps:

1. Establish an FTP session to the FTP server.
2. Issue a PWD command. This should show the TSO prefix as your new default working directory. For information on using the PWD command, see [“PWD subcommand—Display the current working directory” on page 239](#).

Notes:

1. To use the TSO PREFIX as your default working directory, you must have installed RACF Version 1.9 and you must define your TSO user IDs through RACF.
2. When you log in to an FTP server from the z/OS UNIX shell, the default local working directory is the directory from which the FTP client was started.

Obtaining status and system information

To retrieve and display status information about the local host and remote host, use the subcommands listed in [Table 7 on page 33](#).

Table 7. FTP subcommands for obtaining status and system information

| Subcommand | Description | See |
|------------|--|--|
| ! | Passes a z/OS UNIX System Services command to the local z/OS shell. This command must be issued while using FTP in the z/OS shell. | “! subcommand—Invoke a z/OS UNIX System Services function” on page 143 |
| DEBug | Sets general trace options. | “DEBug subcommand—Set general trace options” on page 154 |
| DUMP | Sets extended trace options. | “DUMP subcommand—Set extended trace options” on page 163 |

Table 7. FTP subcommands for obtaining status and system information (continued)

| Subcommand | Description | See |
|------------|---|--|
| FEature | Asks server which features or extensions it supports. | “FEature subcommand—Query FTP server for features it supports” on page 166 |
| LOCStat | Displays FTP status information for the local host. | “LOCStat subcommand—Display local status information ” on page 201 |
| NOop | Checks whether the foreign host is still responding. | “NOop subcommand—Test the connection ” on page 230 |
| STatus | Displays status information for the foreign host. | “STatus subcommand—Retrieve status information from a remote host” on page 278 |
| SYstem | Displays the name of the foreign host’s operating system. | “SYstem subcommand—Display the operating system name ” on page 287 |
| TSO | Passes a TSO command to the local host TSO environment. | “TSO subcommand—Use TSO commands ” on page 288 |

Working with directories on the remote host

To obtain directory information on the remote host, use the subcommands listed in [Table 8 on page 34](#).

Table 8. FTP subcommands for working with directories on the remote host

| Subcommand | Description | See |
|------------|--|---|
| CD | Changes the working directory. | “CD subcommand—Change the directory on the remote host ” on page 149 |
| CDUp | Changes to the parent of the current working directory. | “CDUp subcommand—Change to the parent of the working directory” on page 151 |
| CWd | Changes the working directory (synonymous with CD). | “CD subcommand—Change the directory on the remote host ” on page 149 |
| DIr | Lists the directory entries for files on the foreign host. | “DIr subcommand—Obtain a list of directory entries ” on page 159 |
| LS | Lists the names of files on the foreign host. | “LS subcommand—Obtain a list of file names ” on page 208 |
| MKdir | Creates a directory on the foreign host. | “MKdir subcommand—Create a directory on the remote host ” on page 216 |
| PWd | Displays the name of the active working directory on the foreign host. | “PWd subcommand—Display the current working directory ” on page 239 |
| RMdir | Removes a directory on the foreign host. | “RMdir subcommand—Remove a directory on the remote host” on page 244 |

Examples of the CD subcommand

This example shows how to change and choose remote working directories and how the z/OS FTP server enables you to switch between the MVS and z/OS UNIX file system environments. For more information on how to change the directory levels, see the information on the CD, CDUP, and LCD subcommands.

```

User: ftp 9.67.113.24 621
System:
    IBM FTP CS V1R5
    FTP: using TCPCS
    Connecting to 9.67.113.24, port 621
    220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
    220 Connection will not timeout.
    USER(identify yourself to the host):
    NAME (<host>:tsouserid):
User: user121
System:
    >>>USER user121
    331 Send password please.
    Password:

    >>>PASS *****
    230 USER121 is logged on. Working directory is "/u/user121".
    Command:
User: cd tcip
System:
    >>>CWD tcip
    250 HFS directory /u/user121/tcip is the current working directory
    Command:
User: cd ..
System:
    >>>CWD ..
    250 HFS directory /u/user121 is the current working directory
    Command:
User: cd 'user121'
System:
    250 "'user121'" is working directory name prefix.
    Command:

```

Examples showing the differences between DIR and LS output for z/OS UNIX directories

The examples in this section use the following Internet addresses:

MVSXA2: 9.67.113.25

MVSXA3: 9.67.113.24

The current host is MVSXA2 (9.67.113.25). An FTP command is issued from 9.67.113.25 to 9.67.113.24.

```

User: ftp 9.67.113.24
System:
    IBM FTP CS V1R5
    FTP: using TCPCS
    Connecting to 9.67.113.24, port 621
    220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
    220 Connection will not timeout.
    NAME (<host>:tsouserid):
User: user121
System:
    >>>USER user121
    331 Send password please.
    Password:

    >>>PASS *****
    230 USER21 is logged on. Working directory is "/u/user121".
    Command:

```

```

User: dir
System:
>>>PORT 9,67,112,25,4,25
200 Port request OK.
>>>LIST
125 List started OK.
total 2736
drwxr-xr-x      2 USER121 SYS1      0 Nov 20 18:15 IBM
-rwxr-xr-t      2 USER121 SYS1 389120 Feb  5 16:03 ftpdka
-rwxr-xr-t      2 USER121 SYS1 962560 Feb  5 16:04 ftpsrvka
-rw-r-----    1 USER121 SYS1  11648 Jan 20 14:30 g.s
drwxr-x---      3 USER121 SYS1      0 Oct 21 17:50 msg
-rw-r-----    1 USER121 SYS1  1458 Jan 10 19:25 s.k
drwxr-x---      2 USER121 SYS1      0 Feb  6 15:59 tcpip
drwxr-x---      2 USER121 SYS1      0 Feb  6 17:29 test
250 List completed successfully.

Command:
User: ls
System:
>>>PORT 9,67,112,25,4,26
200 Port request OK.
>>>NLST
125 List started OK.
IBM
ftpdka
ftpsrvka
g.s
msg
s.k
tcpip
test
250 List completed successfully.
Command:

```

DIR provides detailed information about the data sets under the remote working directory, while LS shows the data set names only.

Examples showing the differences between DIR and LS output with DIRECTORYMode and DATASetmode for MVS

This section gives examples of issuing a DIR and LS command in both DIRECTORYMode and DATASetmode.

```

User: ftp 1.1.2.3
System:
IBM FTP CS V1R5
FTP: using TCPCS
Connecting to 1.1.2.3, port 21
220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
220 Connection will close if idle for more than 5 minutes.
NAME (<host>:tsouserid):

User: mvsuser
System:
>>>USER mvsuser
331 Send password please.
Password:
>>>PASS *****
230 MVSUSER is logged on. Working directory is "/u/mvsuser"
Command:

```

```

User: cd 'tcpv3'
System:
    >>>CWD 'tcpv3'
    257 "'TCPV3.'" is working directory name prefix.
    Command:
User: site directorymode
System:
    >>>SITE directorymode
    200 Site command was accepted
    Command:
User: dir
System:
    >>>PORT 1,1,2,2,4,39
    200 Port request OK.
    >>>LIST 125 List started OK.
    Volume Unit
    Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    Pseudo Directory
    250 List completed successfully.
    Command:

```

ETC
FTP
HOSTS
NSMAIN
PROFILE
STANDARD
TCPIP
TCPIPL62
TELNET

```

User: site datasetmode
System:
    >>>SITE datasetmode
    200 Site command was accepted
    Command:
User: dir
System:
    >>>PORT 1,1,2,2,4,40
    200 Port request OK.
    >>>LIST
    125 List started OK.
    Volume Unit      Date  Ext  Used  Recfm  Lrecl  BlkSz  Dsorg  Dsname
    APCSPL 3380D 07/16/97 1 1 FB 80 8800 PS ETC.RPC
    APCSPL 3380D 08/03/97 1 1 FB 80 3200 PS ETC.SERVICES
    APCSPL 3380D 08/03/97 1 1 FB 80 3120 PS FTP.DATA
    APCSPL 3380D 08/02/97 1 1 F 158 158 PS HOSTS.ADDRINFO
    APCSPL 3380D 08/03/97 1 1 FB 80 3120 PS HOSTS.LOCAL
    APCSPL 3380D 07/30/97 1 1 F 56 56 PS HOSTS.SITEINFO
    APCSPL 3380D 07/15/97 1 1 FB 80 8800 PS NSMAIN.CACHE
    APCSPL 3380D 07/28/97 1 1 FB 80 8800 PS NSMAIN.DATA
    APCSPL 3380D 08/03/97 1 2 FB 80 3200 PS PROFILE.TCPIP
    APCSPL 3380D 07/26/97 1 2 FB 80 3200 PS PROFILE.TCPIP.XA2
    APCSPL 3380D 08/03/97 1 1 VB 5124 6160 PS STANDARD.TCPKJBIN
    APCSPL 3380D 08/03/97 1 15 F 256 256 PS STANDARD.TCPXLBIN
    APCSPL 3380D 08/03/97 1 1 FB 80 3120 PS TCPIP.DATA
    APCSPL 3380D 06/29/97 1 2 FB 80 3200 PS TCPIPL62.CONFIG
    APCSPL 3380D 07/29/97 1 15 F 256 256 PS TELNET.TCPXLBIN
    250 List completed successfully.
    Command:

```

```

User: site directorymode
System:
    >>>SITE directorymode
    200 Site command was accepted

    Command:
User: ls
System:
    >>>PORT 1,1,2,2,4,41
    200 Port request OK.
    >>>NLST
    125 List started OK.
    ETC
    FTP
    HOSTS
    NSMAIN
    PROFILE
    STANDARD
    TCPIP
    TCPIPL62
    TELNET
    250 List completed successfully.

    Command:

```

```

User: site datasetmode
System:
    >>>SITE datasetmode
    200 Site command was accepted

    Command:
User: ls
System:
    >>>PORT 1,1,2,2,4,42
    200 Port request OK.
    >>>NLST
    125 List started OK.
    ETC.RPC
    ETC.SERVICES
    FTP.DATA
    HOSTS.ADDRINFO
    HOSTS.LOCAL
    HOSTS.SITEINFO
    NSMAIN.CACHE
    NSMAIN.DATA
    PROFILE.TCPIP
    PROFILE.TCPIP.XA2
    STANDARD.TCPKJBIN
    STANDARD.TCPXLBIN
    TCPIP.DATA
    TCPIPL62.CONFIG
    TELNET.TCPXLBIN
    250 List completed successfully.

    Command:

```

Working with directories on the local host

To work with directories on the local host, use the subcommands listed in [Table 9 on page 38](#).

Table 9. FTP subcommands for working with directories on the local host

| Subcommand | Description | See |
|------------|--|---|
| LCd | Changes the current directory on the local host. | “LCd subcommand—Change the local working directory” on page 175 |
| LMkdir | Creates a directory on the local host. | “LMkdir subcommand—Create a directory on the local host” on page 177 |
| LPwd | Displays the name of the active working directory on the local host. | “LPwd subcommand—Display the current working-level qualifier” on page 208 |

Your default working directory on the local host is set according to the environment in which the FTP client is invoked: \$HOME in z/OS UNIX, your MVS user ID in TSO.

The following examples show how to choose local working directories.

```
User: ftp 1.1.2.3
System:
      IBM FTP CS V1R5
      FTP: using TCPCS
      Connecting to 1.1.2.3, port 21
      220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
      220 Connection will close if idle for more than 5 minutes.
      NAME (<host>:tsouserid):
User: mvsuser
System:
      >>>USER mvsuser
      331 Send password please.
      Password:
      >>>PASS *****
      230 MVSUSER is logged on. Working directory is "/u/mvsuser".
      Command:
```

```
User: lpwd
System:
      Local directory is MVSUSER.
      Command:
User: lcd tcpip
System:
      Local directory name set to MVSUSER.TCPIP.
      Command:
User: lpwd
System:
      Local directory is MVSUSER.TCPIP.
      Command:
User: lcd 'ftp.test'
System:
      Local directory name set to FTP.TEST.
      Command:
User: lpwd
System:
      Local directory is FTP.TEST.
      Command:
User: lcd ..
System:
      Local directory name set to FTP.
      Command:
User: lpwd
System:
      Local directory is FTP.
      Command:
```

Security issues when using FTP

The following information describes security issues to consider when using FTP.

Using security mechanisms

File data transferred between an FTP client and server can be secured with respect to encryption, authentication, and data integrity.

Authentication established using a security mechanism can also be used to make the authorization decision. The FTP security interaction begins with a client telling the server what security mechanism it wants to use with the AUTH command. The server either accepts this mechanism, rejects this mechanism, or, in the case of a server which does not implement the security extensions, rejects the

command completely. The server's reply indicates if the client must respond with additional data for the security mechanism to interpret.

Once a security association is established, authentication (which is part of this association) can be used in addition to the standard user ID/password exchange for authorizing a user to connect to the server. A user ID specified by the USER command is always required to specify the identity to be used on the server.

Transport Layer Security (TLS) is an upwardly-compatible successor to Secure Sockets Layer (SSL). SSL is a protocol that performs secure and encrypted TCP transmission. The FTP client supports either SSL or TLS protected sessions, including client authentication. Note that the negotiation of SSL versus TLS is performed by the sockets-layer TLS code and is transparent to FTP.

Many TLS/SSL applications work by having a client connect to one TCP port for unprotected sessions and a separate TCP port for protected ones. FTP supports this mode for compatibility with the original SSL design. However, FTP also provides a more general solution for FTP security, where the client connects to the FTP server on the regular, non-encrypted port and negotiates authentication and encryption options.

FTP assumes that the port configured by the TLSPORT statement (default TLSPORT is 990) is a protected port. An AUTH command is not needed and the client completes the exchange of additional data with the server immediately after a successful connection.

FTP support for SSL/TLS protected sessions is based on the Internet Draft, *On Securing FTP with TLS*. As of October, 2005, *On Securing FTP with TLS* has been published as RFC 4217. The RFC level is different from the Internet draft. You can set the TLSRFCLEVEL configuration option to choose which level of *On Securing FTP with TLS* you want FTP to support.

To use the new RFC 4217 or CCCNONOTIFY function, the client and server must have the same TLSRFCLEVEL value.

Table 10 on page 40 identifies important differences between the draft level and the RFC level of the TLSRFCLEVEL value.

| Table 10. Important differences between the draft, RFC, and CCCNONOTIFY levels | | |
|--|--|---|
| Draft level | RFC level | CCCNONOTIFY level |
| The server does not support the CCC and AUTH commands when the connection is secured with TLS. | The server supports the CCC and AUTH commands on connections that are secured with TLS but not implicitly secured by connecting to the port that is configured with the TLSPORT statement. | The server supports the CCC command but not the AUTH command on connections that are secured with TLS but not implicitly secured by connecting to the port that is configured with the TLSPORT statement. The server does not issue a TLSshutdown command when it receives the CCC command. |
| The client does not allow CCc or CProtect clear subcommands during a session that is secured with TLS. | The client allows the CCc and CProtect clear subcommands during a session that is secured with TLS but not implicitly secured by connecting to the port that is configured with the TLSPORT statement. | The client allows the CCc and CProtect clear subcommands during a TLS-secured session that is not implicitly secured by connecting to the port that is configured with the TLSPORT statement. The client does not issue a TLSshutdown command when it sends the CCC command. |
| The client does not allow an AUTH command to flow during a session that is secured with TLS. | The client allows the AUTH command during a session that is secured with TLS but not implicitly secured by connecting to the port that is configured with the TLSPORT statement. | The client does not allow an AUTH command to flow during a session that is secured with TLS. |

The server does not allow the PROT or PBSZ commands to run on a control connection that has been cleared with the CCC command. After a successful CCC subcommand, the client does not allow clear, protect clear, private, and protect private subcommands for the remainder of the session.

There are optional FTP commands and statements for negotiating session security. The following list of the configuration parameters determines whether the client uses a security mechanism to protect the session:

- See [“FTP command — Entering the FTP environment ” on page 19](#) for a description of the start parameters.

```
Start parameters
-a    TLS
-a    GSSAPI
-i    TLS
-i    GSSAPI
```

- See [“Changing local site defaults using FTP.DATA” on page 62](#) and the [z/OS Communications Server: IP Configuration Reference](#) for a description of the FTP.DATA statements.

```
FTP.DATA statements
SECURE_MECHANISM
SECURE_FTP
SECURE_CTRLCONN
SECURE_DATACONN
SECURE_SESSION_REUSE
SECUREIMPLICITZOS
CIPHERSUITE
KEYRING
TLSPOPT
TLSTRFCLEVEL
TLSTIMEOUT
SECURE_PBSZ
```

Using a SOCKS server

You can configure the FTP client to access FTP servers through a SOCKS server. When entering FTP through a SOCKS server, the FTP client establishes a connection with the SOCKS server and then the SOCKS server establishes a connection to the FTP server. All data and commands for the FTP session are relayed through the SOCKS server. A SOCKS server can be configured to log all FTP connections and permit or deny access to certain FTP servers. See [“Configuring the FTP client for SOCKS server” on page 63](#) for more information.

FTP client security user exits

The FTP client provides two user exits you can use to restrict commands the FTP client sends to the server, and to monitor FTP replies sent from the server to the client.

User exit EZAFCMD is called for every command the FTP client sends to the server. A user exit routine you write for user exit EZAFCMD can inspect an FTP command and its arguments, modify the arguments of an FTP command, prevent an FTP command from being sent to the server, or end the FTP client before the client sends the command to the server.

User exit EZAFCREP is called for every reply the FTP client receives from the server. For replies comprised of more than one line, EZAFCREP is called once for each line of the reply, as each line is received. A user exit routine you write for EZAFCREP can inspect the FTP server reply, or end the FTP client after the FTP client receives a certain line of the reply sent from the server.

For more information about FTP client security exits, see [Configuring the optional FTP user exits in z/OS Communications Server: IP Configuration Guide](#) and [FTP client user exits in z/OS Communications Server: IP Configuration Reference](#).

Chapter 4. Transferring data using the File Transfer Protocol (FTP)

The FTP command enables you to transfer data sets between your local host and any host that supports TCP/IP and FTP. Using the FTP command and its subcommands, you can sequentially access multiple hosts without leaving the FTP environment.

This topic describes information about:

- [“Preparing the environment for FTP” on page 43](#)
- [“Transferring data with FTP” on page 45](#)
- [“Changing local site defaults using FTP.DATA” on page 62](#)
- [“Sample FTP.DATA data set \(FTCDATA\)” on page 63](#)
- [“Support for SBCS languages” on page 77](#)
- [“FTP with traditional DBCS support” on page 78](#)
- [“Support for MBCS languages” on page 81](#)
- [“Specifying values for new data sets” on page 82](#)
- [“Generation data group support” on page 85](#)
- [“Submitting FTP requests in batch” on page 88](#)
- [“Using the EXEC interface” on page 92](#)
- [“FTP return codes” on page 95](#)
- [“Restarting a failed data transfer” on page 103](#)
- [“Using z/OS UNIX System Services named pipes” on page 104](#)
- [“Interfacing with JES” on page 114](#)
- [“Performing Db2 SQL queries with FTP” on page 130](#)
- [“SUBSYS: Writing to BatchPipes” on page 135](#)

Preparing the environment for FTP

You can use the subcommands that are listed in [Table 11 on page 43](#) to prepare the environment before working with data.

Table 11. FTP subcommands for preparing the environment

| Subcommand | Description | See |
|------------|---|--|
| AScii | Sets the transfer type to ASCII. | “AScii subcommand—Change the data transfer type to ASCII ” on page 146 |
| BINary | Sets the transfer type to IMAGE. | “BINary subcommand—Change the data transfer type to Image ” on page 147 |
| BLOCK | Sets the data transfer mode to block mode. | “BLOCK subcommand—Set the block data transfer mode” on page 148 |
| COMpress | Sets the data transfer mode to compressed mode. | “COMpress subcommand—Set the compressed data transfer mode” on page 153 |
| EBCdic | Sets the transfer type to EBCDIC. | “EBCdic subcommand—Change the data transfer type to EBCDIC ” on page 164 |

Table 11. FTP subcommands for preparing the environment (continued)

| Subcommand | Description | See |
|------------|---|---|
| File | Sets the file structure to File . | “File subcommand—Set the file structure to File” on page 166 |
| GLob | Toggles globbing (the expansion of metacharacters in file names) for the MDELETE, MGET, and MPUT subcommands. | “GLob subcommand—Toggle expansion of metacharacters” on page 168 |
| LOCSite | Specifies information that is used by the local host to provide service specific to that host system. | “LOCSite subcommand—Specify site information to the local host” on page 179 |
| LANGuage | Sets language for server replies. | “LANGuage subcommand—Set the language used for FTP replies from the server” on page 174 |
| MOfde | Specifies the mode or data format of the transfer. | “MOfde subcommand—Set the data transfer mode ” on page 220 |
| PROMpt | Toggles interactive prompting for MDELETE, MGET, and MPUT commands. | “PROMpt subcommand—Toggle interactive prompting for M* commands” on page 234 |
| QUOte | Sends an uninterpreted string of data. | “QUOte subcommand—Send an uninterpreted string of data ” on page 240 |
| RECOrd | Sets the file structure to record. | “RECOrd subcommand—Set the file structure to record” on page 241 |
| SENDPort | Enables or disables automatic transmission of the FTP server PORT command. | “SENDPort subcommand—Toggle the sending of port information ” on page 245 |
| SENDSite | Enables or disables automatic transmission of the SITE subcommand. | “SENDSite subcommand—Toggle the sending of site information ” on page 246 |
| SIte | Sends information to the foreign host using site-specific commands. | “SIte subcommand—Send site-specific information to a host ” on page 247 |
| STREam | Sets the data transfer mode to stream mode. | “STREam subcommand—Set the stream data transfer mode” on page 285 |
| STRucture | Sets the file transfer structure. | “STRucture subcommand—Set the file structure ” on page 286 |
| SUnique | Changes the storage methods. | “SUnique subcommand—Changes the storage method ” on page 286 |
| TYpe | Specifies the transfer type. | “TYpe subcommand—Set the data transfer type ” on page 289 |

Transferring data with FTP

You can use the subcommands listed in [Table 12 on page 45](#) to work with and transfer data.

Table 12. FTP subcommands for transferring data

| Subcommand | Description | See |
|------------|--|--|
| APpend | Appends a data set on your local host to a file on the foreign host. | “APpend subcommand—Append a local data set ” on page 144 |
| DELEte | Deletes a single file on the foreign host. | “DELEte subcommand—Delete files ” on page 158 |
| DELImit | Displays the delimiter character between the <i>file_name</i> and <i>file_type</i> . | “DELImit subcommand—Display the file name delimiter ” on page 158 |
| Get | Copies a file from the foreign host to your local host. | “Get subcommand—Copy files ” on page 166 |
| MDelete | Deletes multiple files on the foreign host. | “MDelete subcommand—Delete multiple files ” on page 211 |
| MGet | Copies multiple files from the foreign host to your local host. | “MGet subcommand—Copy multiple files ” on page 213 |
| MPut | Copies multiple files on your local host to the foreign host. | “MPut subcommand—Copy multiple data sets to the remote host ” on page 221 |
| MVSGet | Copies a remote z/OS data set into a local z/OS data set with the remote data set attributes | “MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes” on page 223 |
| MVSPut | Copies a local z/OS data set into a remote z/OS data set with the local data set attributes | “MVSPut subcommand – Copy a local data set into a remote data set name with the local data set attributes” on page 227 |
| PUt | Copies a file on your local host to the foreign host. | “PUt subcommand—Copy data sets to the remote host ” on page 238 |
| REName | Renames a file on the foreign host. | “REName subcommand—Rename files ” on page 242 |
| REStart | Restarts a checkpointed data transfer. | “REStart subcommand - Restart a checkpointed data transfer” on page 242 |
| SRestart | Restarts a stream mode data transfer. | “SRestart subcommand—Restart a stream data transfer” on page 276 |

How to transfer data with FTP

FTP supports only the data transfer of a data set or file structured as a continuous sequence of data bytes. This ensures that the correct record format is preserved across MVS hosts. Information could be lost or altered during transmission if you use an incorrect transfer.

[Table 13 on page 46](#) shows how to set the transmission attributes for different host systems. For example, VM or MVS host systems use EBCDIC for internal character representation. A text file of ASCII data type contains displayable characters; a carriage return (X'0D') and line feed (X'0A') are used to delimit a line. A text file of EBCDIC data type contains displayable characters; the new-line character (X'15') is used to delimit a line. A binary file contains a contiguous stream of bits with no line delimiters.

Table 13. Recommended methods for data transfer

| Transfer between host types | Data transfer type | Data transfer mode |
|--------------------------------------|---|--------------------|
| EBCDIC and EBCDIC — DBCS text data | IBMKANJI (EBCDIC) | Stream |
| EBCDIC and EBCDIC — text data | EBCDIC | Stream |
| EBCDIC and EBCDIC — DBCS binary data | IBMKANJI (EBCDIC) | Block |
| EBCDIC and EBCDIC — binary data | EBCDIC | Block |
| EBCDIC and ASCII — DBCS text data | SJISKANJI, EUCKANJI, JIS78KJ, JIS83KJ, HANGEUL, KSC5601, TCHINESE, BIG5, SCHINESE (ASCII) | Stream |
| ASCII and EBCDIC — MBCS data | ASCII | Stream |
| ASCII and EBCDIC — text data | ASCII | Stream |
| ASCII and EBCDIC — DBCS binary data | Image (binary) | Stream |
| ASCII and EBCDIC — binary data | Image (binary) | Stream |
| ASCII-to-EBCDIC-to-ASCII — all data | Image (binary) | Stream |

Note: The EBCDIC host is used for storage only. Data remains encoded in ASCII; therefore, the data cannot be used on the EBCDIC host.

For more information about the DBCS data type keywords and examples, see [“FTP with traditional DBCS support”](#) on page 78.

For information about setting data transfer type, see [“TYPe subcommand—Set the data transfer type”](#) on page 289. For information about setting data transfer mode, see [“MODE subcommand—Set the data transfer mode”](#) on page 220.

Examples of Get, MGet and MVSGet subcommands

Results:

1. If the LISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the server FTP.DATA file, the default is as if the LISTSUBdir option was specified on the SITE subcommand.
2. If the z/OS FTP server has the NOLISTSUBDIR option on the SITE subcommand or LISTSUBDIR FALSE in the server FTP.DATA file, then an mget * command gets only the files in the current directory.

Restrictions:

1. The LISTSUBDIR statement applies to z/OS UNIX file operations only. MVS data set operations are not affected.
2. The SITE LISTSUBDIR command is supported by z/OS FTP in V1R7 and later releases. The FTP client must be communicating with a z/OS V1R7 or later FTP server or an unrecognized parameter response results.

Example 1: GET and MGET enable you to obtain files from a remote host and send them to the local host. In this example, FTP subcommands are issued from MVSXA2 to MVSVIC03. See [Table 12 on page 45](#) for other subcommands useful for working with and transferring data.

The following members exist in the data set USER121.FTP.EXAMPLE on MVSVIC03:

FILE1

FILE2
FILE3
FILE4
FILE5

The following is displayed when entering the FTP environment:

```
User: ftp 9.67.113.24 621
System:      IBM FTP CS V1R5
             FTP: using TCP/CS
             Connecting to 9.67.113.24, port 621
             220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
             220 Connection will not timeout.
             NAME (<host>:tsouserid):
User: user121
System:      >>>USER user121
             331 Send password please.
             Password:
             >>>PASS *****
             230 USER121 is logged on. Working directory is "/u/user121".
             Command:

User: get 'user121.ftp.example(file1)' 'user121.ftp.example(file1)'
System:      'USER121.FTP.EXAMPLE(FILE1)' IS AN non-EXISTENT PARTITIONED DATASET.
             USE LMKDIR TO CREATE IT. LOCAL FILE NOT FOUND
             COMMAND:
User: lmkdir 'user121.ftp.example'
System:      USER121.FTP.EXAMPLE CREATED.
             COMMAND:
User: get 'user121.ftp.example(file1)' 'user121.ftp.example(file1)'
System:      >>>PORT 9,67,112,25,4,9
             200 Port request OK.
             >>>RETR 'USER121.ftp.example(file1)'
             125 Sending data set USER121.FTP.EXAMPLE(FILE1) FIXrecfm 128
             250 Transfer completed successfully.
             3464 bytes transferred in 0.754 seconds. Transfer rate 4.59 Kbytes/sec.
             Command:
User: get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)'
System:      >>>PORT 9,67,112,25,4,34
             200 Port request OK.
             >>>RETR 'USER121.ftp.example(file2)'
             125 Sending data set USER121.FTP.EXAMPLE(FILE2) FIXrecfm 128
             250 Transfer completed successfully.
             3464 bytes transferred in 1.483 seconds. Transfer rate 2.34 Kbytes/sec.
             Command:
```

```

User: get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)'
System:
    Data set 'USER121.FTP.EXAMPLE(FILE2)' was not replaced.
    Local file already exists
    To replace it, use command with the (REPLACE option
    Command:
User: get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)' (replace
System:
    >>>PORT 9,67,112,25,4,35
    200 Port request OK.
    >>>RETR 'user121.ftp.example(file2)'
    125 Sending data set USER121.FTP.EXAMPLE(FILE2)
    250 Transfer completed successfully.
    3464 bytes transferred in 0.767 seconds. Transfer rate 0.50 Kbytes/sec.
    Command:
User: lpwd
System:
    Local directory is USER121
    COMMAND:
User: mget 'user121.ftp.example(file3)' 'user121.ftp.example(file4)'
System:
    >>>PORT 9,67,112,25,4,10
    200 Port request OK.
    >>>NLST 'user121.ftp.example(file3)'
    125 List started OK.
    250 List completed successfully.
    >>>PORT 9,67,112,25,4,11
    200 Port request OK.
    >>>NLST 'user121.ftp.example(file4)'
    125 List started OK.
    250 List completed successfully.
    >>>PORT 9,67,112,25,4,12
    200 Port request OK.
    >>>RETR 'USER121.FTP.EXAMPLE(FILE3)'
    125 Sending data set USER121.FTP.EXAMPLE(FILE3)
    250 Transfer completed successfully.
    3993 bytes transferred in 0.745 seconds. Transfer rate 0.51 Kbytes/sec.
    >>>PORT 9,67,112,25,4,13
    200 Port request OK.
    >>>RETR 'USER121.FTP.EXAMPLE(FILE4)'
    125 Sending data set USER121.FTP.EXAMPLE(FILE4)
    250 Transfer completed successfully.
    7367 bytes transferred in 0.818 seconds. Transfer rate 9.01 Kbytes/sec.
    Command:

```

```

User: lpwd
System: Local directory is USER121.
        Command:
User: cd 'user121.ftp.example'
System: >>>CWD 'user121.ftp.example'
        250 "USER121.FTP.EXAMPLE" partitioned data set is working directory.
        Command:
User: pwd
System: >>>PWD
        257 "USER121.FTP.EXAMPLE" partitioned data set is working directory.
        Command:
User: mget file3 file4
System: >>>PORT 9,67,112,25,4,20
        200 Port request OK.
        >>>NLST file3
        125 List started OK.
        250 List completed successfully.
        >>>PORT 9,67,112,25,4,21
        200 Port request OK.
        >>>NLST file4
        125 List started OK.
        250 List completed successfully.
        >>>PORT 9,67,112,25,4,22
        200 Port request OK.
        >>>RETR FILE3
        125 Sending data set USER121.FTP.EXAMPLE(FILE3)
        250 Transfer completed successfully.
        3993 bytes transferred in 0.549 seconds. Transfer rate 0.46 Kbytes/sec.
        >>>PORT 9,67,112,25,4,23
        200 Port request OK.
        >>>RETR FILE4
        125 Sending data set USER121.FTP.EXAMPLE(FILE4)
        250 Transfer completed successfully.
        7367 bytes transferred in 0.936 seconds. Transfer rate 0.23 Kbytes/sec.
        Command:
User: quit
System: >>>QUIT
        221 Quit command received. Goodbye.
        READY

```

```

User: ftp 9.67.113.24 621
System: IBM FTP CS V1R5 2003 314 01:11 UTC
        Connecting to 9.67.113.24, port 621
        220-FTPD1 IBM FTP CS/390 V2R10 AT MVS164, 20:12:38 ON 2003-01-02.
        220 Connection will not timeout.
        NAME (<host>:tsouserid):
User: user121
System: >>>USER user121
        331 Send password please.
        Password:
        >>>PASS *****
        230 USER121 is logged on. Working directory is "/u/user121".
        Command:

```

```

User: get '/u/user121/ftp.example/file1' 'user121.ftp.example(file1)'
System:
    >>>PORT 9,67,112,25,4,24
    200 Port request OK.
    >>>RETR '/u/user121/ftp.example/file1'
    125 Sending data set /u/user121/ftp.example/file1
    250 Transfer completed successfully.
    3464 bytes transferred in 1.391 seconds. Transfer rate 2.49 Kbytes/sec.
    Command:
User: lcd 'user121.ftp.example'
System: Local directory name set to partitioned data set USER121.FTP.EXAMPLE.
    Command:
User: lpwd
System: Local directory is partitioned data set USER121.FTP.EXAMPLE.
    Command:
User: cd '/u/user121/ftp.example'
System: >>>CWD '/u/user121/ftp.example'
    250 HFS directory /u/user121/ftp.example is the current working
        directory
    Command:
User: pwd
System: >>>PWD
    257 "/u/user121.ftp.example" is the HFS working directory.
    Command:
User: get file1
System: >>>PORT 9,67,112,25,4,26
    200 Port request OK.
    >>>RETR file1
    125 Sending data set /u/user121/ftp.example/file1
    250 Transfer completed successfully.
    3464 bytes transferred in 1.059 seconds. Transfer rate 3.27 kbytes/sec.
    Command:

```

```

User: mget '/u/user121/ftp.example/file4' '/u/user121/ftp.example/file5'
System:
    >>>PORT 9,67,112,25,4,33
    200 Port request OK.
    >>>NLST '/u/user121/ftp.example/file4'
    125 List started OK
    250 List completed successfully.
    >>>PORT 9,67,112,25,4,34
    200 Port request OK.
    >>>NLST '/u/user121/ftp.example/file5'
    125 List started OK
    250 List completed successfully.
    >>>PORT 9,67,112,25,4,35
    200 Port request OK.
    >>>RETR /u/user121/ftp.example/file4
    125 Sending data set /u/user121/ftp.example/file4
    250 Transfer completed successfully.
    7367 bytes transferred in 1.324 seconds. Transfer rate 5.56
        kbytes/sec.
    200 Port request OK.
    >>>RETR /u/user121/ftp.example/file5
    125 Sending data set /u/user121/ftp.example/file5
    250 Transfer completed successfully.
    3464 bytes transferred in 0.951 seconds. Transfer rate 3.64
        kbytes/sec.
    Command:

```

The data set USER121.FTP.EXAMPLE on MVSXA2 now contains the following members:

```

FILE1
FILE2
FILE3
FILE4
FILE5

```

Restrictions:

1. You do not have a choice of names for the local file as a result of the MGET subcommand.
2. The MGET subcommand is not applicable for generation data groups (GDGs).

Example 2: MGET with SITE LISTSUBDIR

Following is an example of `mget *` with `SITE LISTSUBDIR`. This setting affects processing of the `NLST` command. The z/OS FTP client sends an `NLST` command to the server as part of `mget *` subcommand processing. `LISTSUBDIR` specifies that both the current and next subdirectory should be retrieved from the server as a result of processing an `mget *` subcommand. In this example, the current directory has a file `x` and a subdirectory `y` and subdirectory `y` has a file `x`.

```

site listsubdir
>>> SITE listsubdir
200 SITE command was accepted
mget * (rep
>>> PORT 127,0,0,1,4,13
200 Port request OK.
>>> NLST *
125 List started OK
250 List completed successfully.
>>> PORT 127,0,0,1,4,14
200 Port request OK.
>>> RETR x
125 Sending data set /tmp/mgetmput/x
250 Transfer completed successfully.
>>> PORT 127,0,0,1,4,14
200 Port request OK.
>>> RETR y/x
125 Sending data set /tmp/mgetmput/y/x
250 Transfer completed successfully.
5 bytes transferred in 1.010 seconds. Transfer rate 0.00 Kbytes/sec.
Command:

```

Example 3: MGET with SITE NOLISTSUBDIR

Following is an example of `mget *` with `SITE NOLISTSUBDIR`. This setting affects processing of the `NLST` command. The z/OS FTP client sends an `NLST` command to the server as part of `mget *` subcommand processing. `NOLISTSUBDIR` specifies that only the current directory should be retrieved from the server as a result of processing an `mget *` subcommand. In this example, the current directory has a file `x` and a subdirectory `y` and subdirectory `y` has a file `x`.

```

site nolistsubdir
>>> SITE nolistsubdir
200 SITE command was accepted
mget * (rep
>>> PORT 127,0,0,1,4,13
200 Port request OK.
>>> NLST *
125 List started OK
250 List completed successfully.
>>> PORT 127,0,0,1,4,14
200 Port request OK.
>>> RETR x
125 Sending data set /tmp/mgetmput/x
250 Transfer completed successfully.
5 bytes transferred in 1.010 seconds. Transfer rate 0.00 Kbytes/sec.
Command:

```

Example 4: MVSGET with a physical sequential data set transferred

Following is a sample entry and response that is displayed after the `MVSGET` subcommand is used to transfer a physical sequential data set:

```

mvsget 'user1.ps.source' 'user1.ps.target' (REALlocate
EZA1701I >>> XDSS 'user1.ps.source'
200-LASTREF=2011/12/06 DSEMPY=FALSE
200 SITE DSNTYPE=BASIC RECFM=VB BLKSIZE=6233 LRECL=256 PRIMARY=16 SECONDARY=1 TRACKS EA
TTR=SYSTEM
EZZ9815I local site variables have changed
EZA1701I >>> PORT 127,0,0,1,4,5
200 Port request OK.
EZA1701I >>> RETR 'user1.ps.source'
125 Sending data set USER1.PS.SOURCE
250 Transfer completed successfully.
EZA2108I Confidence=High for MVSGET of USER1.PS.TARGET
EZA1617I 286 bytes transferred in 0.020 seconds. Transfer rate 14.30 Kbytes/sec

```

Restrictions: For more restrictions about the MVSGet subcommand, see [“MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes” on page 223](#)

Example 5: MVSGet with a PDS data set transferred

Following is a sample entry and response that is displayed after the MVSGet subcommand is used to transfer a PDS data set:

```
mvsget 'user1.remote.pds' 'user1.local.pds' (REALlocate
EZA1701I >>> XDSS 'user1.remote.pds'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMARY=1 SECO
NDARY=1 TRACKS EATTR=SYSTEM
EZZ9815I local site variables have changed
EZA2245I "USER1.LOCAL.PDS" created.
EZA2081I Local directory name set to partitioned data set USER1.LOCAL.PDS
EZA1701I >>> PWD
257 "'USER1.'" is working directory.
EZA1701I >>> CWD 'user1.remote.pds'
250 The working directory "USER1.REMOTE.PDS" is a partitioned data set
EZA1701I >>> PORT 127,0,0,1,4,5
200 Port request OK.
EZA1701I >>> NLST *
125 List started OK.
250 List completed successfully.
EZA1701I >>> PORT 127,0,0,1,4,6
200 Port request OK.
EZA1701I >>> RETR NEW1
125 Sending data set USER1.REMOTE.PDS(NEW1)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.010 seconds. Transfer rate 13.40 Kbytes/sec.
EZA1701I >>> PORT 127,0,0,1,4,7
200 Port request OK.
EZA1701I >>> RETR NEW2
125 Sending data set USER1.REMOTE.PDS(NEW2)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.010 seconds.
Transfer rate 13.40 Kbytes/sec.
EZA2581I Local HFS directory is /u/user1 .
EZA1701I >>> CWD 'USER1.'
250 "USER1." is working directory name prefix
EZA2108I Confidence=High for MVSGET of USER1.LOCAL.PDS
```

Restrictions: For more restrictions about the MVSGet subcommand, see [“MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes” on page 223](#)

Examples of PUT, MPUT and MVSPut subcommands

Results:

1. If the LISTSUBdir option is not specified on the LOCSITE subcommand and the LISTSUBDIR statement is not specified in the client FTP.DATA file, the default is as if the LISTSUBdir option was specified on the LOCSITE subcommand.
2. If the z/OS FTP client has the NOLISTSUBDIR option on the LOCSITE subcommand or LISTSUBDIR FALSE in the client FTP.DATA file, an mput * stores only the files that are in the current directory.

Restriction: The LISTSUBDIR statement applies to z/OS UNIX file operations only; MVS data set operations are not affected.

Example 1: PUT and MPUT subcommands enable you to send files from a local host to a remote host. In this example, FTP subcommands are issued from MVSXA2 to MVSVIC03. The data set USER121.FTP.EXAMPLE on MVSXA2 contains the following members:

```
APPEND01
XA2FILE1
XA2FILE2
XA2FILE3
```

The data set USER121.FTP.EXAMPLE on MVSVIC03 contains the following members:

XA3FILE1
XA3FILE2
XA3FILE3

The following is displayed when entering the FTP environment:

```
User: ftp 1.1.2.3
System: IBM FTP CS V1R5
        FTP: using TCPCS
        Connecting to 1.1.2.3, port 21
        220-FTPD1 IBM FTP CS V1R2 at MVS164, 20:12:38 on 2003-01-02.
        220 Connection will close if idle for more than 5 minutes.
        NAME (<host>:tsouserid):
User: user121
System: >>>USER user121
        331 Send password please.
        Password:
        >>>PASS *****
        230 user121 is logged on. Working directory is "/u/user121"
        Command:
```

```
User: put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
System: >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
        200 Site command was accepted
        >>>PORT 1,1,2,2,4,48
        200 Port request OK.
        >>>STOR 'user121.ftp.example(f1from2)'
        125 Storing data set USER121.FTP.EXAMPLE(F1FROM2)
        250 Transfer completed successfully.
        390 bytes transferred in 1.117 seconds.
        Transfer rate 0.35 Kbytes/sec.
        Command:
User: put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
System: >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
        200 Site command was accepted
        >>>PORT 1,1,2,2,4,49
        200 Port request OK.
        >>>STOR 'user121.ftp.example(f1from2)'
        125 Storing data set USER121.FTP.EXAMPLE(F1FROM2)
        250 Transfer completed successfully.
        390 bytes transferred in 0.680 seconds.
        Transfer rate 0.57 Kbytes/sec.
        Command:
```

```
User: sunique
System: Store unique is ON
        Command:
User: put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
System: >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
        200 Site command was accepted
        >>>PORT 1,1,2,2,4,50
        200 Port request OK.
        >>>STOU 'user121.ftp.example(f1from2)'
        125 Storing data set USER121.FTP.EXAMPLE(F1FROM21) ( unique name )
        250 Transfer completed successfully.
        390 bytes transferred in 1.085 seconds.
        Transfer rate 0.36 Kbytes/sec.
        Command:
User: sunique
System: Store unique is OFF
        Command:
User: cd 'user121.ftp.example.'
System: >>>CWD 'user121.ftp.example.'
        257 "USER121.FTP.EXAMPLE." is working directory name prefix.
        Command:
User: lpwd
System: Local directory is USER121.
        Command:
User: lcd 'user121.ftp.example'
System: Local directory name set to PDS USER121.FTP.EXAMPLE.
        Command:
User: lpwd
System: Local directory is partitioned data set USER121.FTP.EXAMPLE.
        Command:
```

```

User:      mput xa2file2 xa2file3
System:    >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
          200 Site command was accepted
          >>>PORT 1,1,2,2,4,51
          200 Port request OK.
          >>>STOR XA2FILE2
          125 Storing data set USER121.FTP.EXAMPLE.XA2FILE2
          250 Transfer completed successfully.
          390 bytes transferred in 1.437 seconds.
          Transfer rate 0.27 Kbytes/sec.
          >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
          200 Site command was accepted
          >>>PORT 1,1,2,2,4,52
          200 Port request OK.
          >>>STOR XA2FILE3
          125 Storing data set USER121.FTP.EXAMPLE.XA2FILE3
          250 Transfer completed successfully.
          390 bytes transferred in 1.091 seconds.
          Transfer rate 0.36 Kbytes/sec.
          Command:
User:      quit
System:    >>>QUIT
          221 Quit command received. Goodbye.
          READY

```

The data set USER121.FTP.EXAMPLE on MVSVIC03 now contains the following members:

```

F1FROM2
F1FROM21
XA3FILE1
XA3FILE2
XA3FILE3

```

MVSVIC03 now also has the following data sets:

```

USER121.FTP.EXAMPLE.XA2FILE2
USER121.FTP.EXAMPLE.XA2FILE3

```

Restriction: The MPUT command is not applicable for generation data groups (GDGs).

Example 2: MPUT with LOCSITE LISTSUBDIR

Following is an example of mput * with the LOCSITE LISTSUBDIR option. The LISTSUBDIR option specifies that not only the current subdirectory, but also the next subdirectory should be searched for files to be sent from the client to the server. In this example, the current directory has a file x and a subdirectory y and subdirectory y has a file x.

```

locsite listsubdir
prompt
Interactive mode is off
Command:
mput *
>>> PORT 127,0,0,1,4,11
200 Port request OK.
>>> STOR x
125 Storing data set /u/user1/x
250 Transfer completed successfully.
5 bytes transferred in 0.070 seconds. Transfer rate 0.07 Kbytes/sec.
>>> PORT 127,0,0,1,4,12
200 Port request OK.
>>> STOR x
125 Storing data set /u/user1/x
250 Transfer completed successfully.
5 bytes transferred in 0.020 seconds. Transfer rate 0.25 Kbytes/sec.
Command:

```

Example 3: MPUT with LOCSITE NOLISTSUBDIR

Following is an example of mput * with the LOCSITE NOLISTSUBDIR option. The NOLISTSUBDIR option specifies that only the current directory should be searched for files to be sent from the client to the server. In this example, the current directory has a file x and a subdirectory y and subdirectory y has a file x.


```

locsite N0listsubdir
prompt
Interactive mode is off
Command:
mput *
>>> PORT 127,0,0,1,4,11
200 Port request OK.
>>> STOR x
125 Storing data set /u/user1/x
250 Transfer completed successfully.
5 bytes transferred in 0.070 seconds. Transfer rate 0.07 Kbytes/sec.
Command:

```

Example 4: MVSPut with a physical sequential data set transferred

Following is a sample entry and response that is displayed after the MVSPut subcommand is used to transfer a physical sequential data set:

```

mvspu 'user1.ps.source' 'user1.ps.target' (REAllocate
EZA1701I >>> XDSS 'user1.ps.target'
200-LASTREF=2011/12/07 DSEMPY=FALSE
200 SITE DSNTYPE=BASIC RECFM=VB BLKSIZE=6233 LRECL=256 PRIMARY=1 SECONDARY=1 TRA
CKS EATTR=SYSTEM
EZA1701I >>> DELE 'user1.ps.target'
250 USER1.PS.TARGET deleted.
EZA1701I >>> SITE DSNTYPE=BASIC RECFM=VB BLKSIZE=6233 LRECL=256 PRIMARY=1 SECOND
ARY=1 TRACKS EATTR=SYSTEM
200 SITE command was accepted
EZA1701I >>> PORT 127,0,0,1,4,4
200 Port request OK.
EZA1701I >>> STOR 'user1.ps.target'
125 Storing data set USER1.PS.TARGET
250 Transfer completed successfully.
EZA2108I Confidence=High for MVSPUT of USER1.PS.TARGET
EZA1617I 2331 bytes transferred in 0.005 seconds. Transfer rate 466.20 Kbytes/sec.

```

Restriction: For more restrictions about the MVSPut subcommand, see [“MVSPut subcommand – Copy a local data set into a remote data set name with the local data set attributes” on page 227](#)

Example 5: MVSPut with a PDS data set transferred

Following is a sample entry and response that is displayed after the MVSPut subcommand is used to transfer a PDS data set:

```

mvspu 'user1.local.pds' 'user1.remote.pds' (REAllocate
EZA1701I >>> PWD
257 "'USER1.'" is working directory.
EZA1701I >>> XDSS 'user1.remote.pds'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMARY=1 SECO
NDARY=1 TRACKS EATTR=SYSTEM
EZA1701I >>> DELE 'user1.remote.pds'
250 USER1.REMOTE.PDS deleted.
EZA1701I >>> SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMA
RY=1 SECONDARY=1 TRACKS EATTR=SYSTEM
200 SITE command was accepted
EZA2081I Local directory name set to partitioned data set USER1.LOCAL.PDS
EZA1701I >>> MKD 'user1.remote.pds'
257 "'USER1.REMOTE.PDS'" created.
EZA1701I >>> CWD 'user1.remote.pds'
250 The working directory "USER1.REMOTE.PDS" is a partitioned data set
EZA1701I >>> PORT 127,0,0,1,4,11
200 Port request OK.
EZA1701I >>> STOR NEW1
125 Storing data set USER1.REMOTE.PDS(NEW1)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.005 seconds. Transfer rate 26.80 Kbytes/sec.
EZA1701I >>> PORT 127,0,0,1,4,12
200 Port request OK.
EZA1701I >>> STOR NEW2
125 Storing data set USER1.REMOTE.PDS(NEW2)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.005 seconds. Transfer rate 26.80 Kbytes/sec.
EZA2581I Local HFS directory is /u/user1.
EZA1701I >>> CWD 'USER1.'
250 "USER1." is working directory name prefix
EZA2108I Confidence=High for MVSPUT of USER1.LOCAL.PDS

```

Restriction: For more restrictions about the MVSPut subcommand, see [“MVSPut subcommand – Copy a local data set into a remote data set name with the local data set attributes”](#) on page 227.

ddname support with FTP

This section describes how the FTP client transfers a data set or file allocated in the JCL for a batch job or by an interactive user prior to the transfer. The FTP client refers to the data set with the ddname used on the allocation.

The FTP Client API does not support ddname transfers. The ddnames associated with a batch job that invokes an application program using the FTP Client API are not available to the created FTP client process.

The //DD: token prefixed before a 1–8 character local file name on a client file access command indicates that the token which follows is actually a ddname, rather than a local file name. This ddname must be allocated by the user (for example, in the JCL that started the FTP client). The server file name must be explicitly specified when a ddname is being used to access a local file for a put command.

Sometimes the client requires DCB information before it opens a data set. Among the situations where this is true are:

- Reading and writing spanned records (RECFM=VS or VBS)
- Reading and writing records that contain ASA control characters
- Reading and writing variable-length records while preserving the RDW
- Reading and writing fixed-length records while preserving trailing blanks

When a data set is allocated using a ddname and the DCB information is needed before open, the FTP client must be able to find the DCB information on the DD statement that was used to allocate the data set.

DCB attributes for a ddname allocation are acquired using the attributes or data set name specified in the DD statement DCB parameter. See the [z/OS MVS JCL Reference](#) for restrictions on using backward references in the DCB parameter.

If the DD statement refers to a cataloged DASD data set, any DCB attributes that are not specified are retrieved from the DSCB. DCB attributes on the DD statement override those found in the DSCB, except that LRECL=0 and BLKSIZE=0 do not override a different value in the DSCB.

If the DD statement refers to a tape data set that is to be opened for input (PUT //DD:), the record format that is specified on the DD statement is used instead of the READTAPEFormat setting. If no record format (RECFM) is specified on the DD statement, the READTAPEFormat setting (if any) is used.

Once the data set is opened by FTP, its attributes are set using the data returned in the DCB by open.

Restrictions:

- If you pass a dynamically allocated ddname to a batch job, do not allocate the ddname using the XTIO, UCB nocapture, or DSAB above the 16 MB line options.
- The MVSGet subcommand does not support specifying the local data set as a ddname.
- To prevent transferring data from an empty file, FTP checks whether the first file in a concatenation series is empty and allocates an empty data set. No data is transferred.

If you use BSAM to transfer a series of concatenated files, each file in the series must not be empty. That is, they must have a valid end of file indicator set.

For concatenated data sets, if any of the files are empty, you can use the PUT and APPEND commands to transfer the data sets. If one of the data sets is empty, the next command continues to run and the additional data is concatenated.

```
put      //dd:infile1 target.ds
append  //dd:infile2 target.ds
append  //dd:infile3 target.ds
```

Following is a sample JCL that illustrates the problem where input consists of concatenated data sets with the first file USER35.GDG1 being empty and the remaining files not being empty. FTP checks an empty data set on the first file, resulting in the transfer of a null file.

```
//STEP02 EXEC PGM=FTP,REGION=2048K,PARM='(TCP TCPCS TRACE'
//STEPLIB DD DSN=USER33.LINKLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//INFILE1 DD DSN=USER35.GDG1(+1),DISP=SHR
//          DD DSN=USER35.GDG2(+1),DISP=SHR
//          DD DSN=USER34.FILE,DISP=SHR
//OUTPUT DD SYSOUT=*
//INPUT DD *
9.67.113.57 21
USER33 **pw**
put //DD:INFILE1 remote.file
quit
/*
```

To resolve this problem, you can use the following sample JCL instead:

```
//STEP02 EXEC PGM=FTP,REGION=2048K,PARM='(TCP TCPCS TRACE'
//STEPLIB DD DSN=USER33.LINKLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//INFILE1 DD DSN=USER35.GDG1(+1),DISP=SHR
//INFILE2 DD DSN=USER35.GDG2(+1),DISP=SHR
//INFILE3 DD DSN=USER34.FILE,DISP=SHR
//OUTPUT DD SYSOUT=*
//INPUT DD *
9.67.113.57 21
USER33 **pw**
put //DD:INFILE1 remote.file
append //DD:INFILE2 remote.file
append //DD:INFILE3 remote.file
quit
/*
```

Note: This restriction applies to all types of data sets, not only GDG data sets.

Following is a sample job that shows usage of the //DD: token. In the sample job there are two data sets that use the local file specification with the //DD: token. One is a data set that is created as a new GDG

data set in STEP01 (see the OUTSET DD statement). Note that STEP02 (the FTP step) uses a backward reference with the DD02 DD statement to locate the data set. Since the referenced DD statement contains explicit DCB attributes, FTP can access the attributes prior to opening the data set. The second data set is an old data set that existed before the job was executed.

```

//USER33J JOB MSGLEVEL=1,MSGCLASS=H,USER=USER33,PASSWORD=**pw**
//STEP01 EXEC PGM=IEBDG
//SYSPRINT DD SYSOUT=A
1 //OUTSET DD DSN=USER33.MYGDG(+1),DISP=(NEW,CATLG,CATLG),
// VOLUME=SER=CPDLB1,SPACE=(TRK,(5,5)),UNIT=SYSDA,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//SYSIN DD *
    < create statements >
//STEP02 EXEC PGM=FTP,REGION=2048K,PARM='(TCP TCPCS TRACE'
//STEPLIB DD DSN=USER33.LINKLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
2 //DD01 DD DSN=USER33.TEST.S.A,DISP=OLD
3 //DD02 DD DSN=*.STEP01.OUTSET,DISP=SHR
//OUTPUT DD SYSOUT=*
//INPUT DD *
9.67.113.57 6321
USER33 **pw**
4 put //DD:DD02 data
5 get data //DD:DD01
quit
/*

```

Following are short descriptions of the numbered items in the example.

- 1 DD statement that allocates a new generation of a GDG data set
- 2 DD statement for an existing data set
- 3 Backward reference for the new data set in STEP01
- 4 Put subcommand using the //DD: token for the new data set created in STEP01
- 5 Get subcommand using the //DD: token for the existing data set

The FTP output for the above job is the following. Note that only a few selected FTP trace statements are shown.

```

EZA1736I FTP (TCP TCPCS
EZA1450I IBM FTP CS V1R5 2003 090 19:22 UTC
EZA1466I FTP: using TCPCS
EZA1456I Connect to ?
EZA1736I 9.67.113.57 6321
EZA1554I Connecting to: 9.67.113.57 port: 6321.
220-FTPDJG1 IBM FTP CS V1R2 at MVS164, 14:58:36 on 2003-01-01.
220 Connection will not timeout.
EZA1459I NAME (9.67.113.57:USER33):
EZA1701I >>> USER USER33
331 Send password please.
EZA1701I >>> PASS
230 USER33 is logged on. Working directory is "/u/user33".
EZA1460I Command:
1 EZA1736I put //DD:DD02 data
2 MF0680 seq_open_file: DDname DD02 has filename USER33.MYGDG.G0087V00
EZA1701I >>> PORT 9,67,113,57,6,158
200 Port request OK.
EZA1701I >>> STOR data
125 Storing data set /u/user33/data
250 Transfer completed successfully.
EZA1617I 820 bytes transferred in 0.020 seconds. Transfer rate 41.00 Kbytes/sec.
EZA1460I Command:
3 EZA1736I get data //DD:DD01
4 MF0680 seq_open_file: DDname DD01 has filename USER33.TEST.S.A
EZA1701I >>> PORT 9,67,113,57,6,159
200 Port request OK.
EZA1701I >>> RETR data
125 Sending data set /u/user33/data

```

```
250 Transfer completed successfully.  
EZA1617I 820 bytes transferred in 0.030 seconds. Transfer rate 27.33 Kbytes/sec.  
EZA1460I Command:  
EZA1736I quit  
EZA1701I >>> QUIT  
221 Quit command received. Goodbye.
```

- 1 Put subcommand using //DD: token for the local file
- 2 Trace statement showing that the local data set name is the new GDG data set that was created in STEP01
- 3 Get subcommand using //DD: token for the local file
- 4 Trace statement showing that the local data set name is the existing data set.

Load module transfer with FTP

As long as your FTP client and FTP server are both at the z/OS Communications Server V2R10 or later, you can use FTP to transfer MVS load modules between load libraries on different hosts or the same host. MVS load modules transferred, using z/OS Communications Server V2R10 or later support, are executable on the target system. A load module can be specified by its real name or by one of its alias names, and in either case, all aliases are transferred with each load module. Load module transfer (LMTR) is also supported for proxy transfer, in which case all three hosts (client, primary server, and secondary server) must be z/OS Communications Server V2R10 or later.

Load module transfer processing (at z/OS V1R2 Communications Server or later) makes use of the IEBCOPY system utility, which must be available on both the origin and destination hosts.

The following FTP file transfer commands will properly transfer MVS load modules:

- get
- mget
- mput
- mvsgget
- mvsgput
- put

Because of the special requirements of MVS load modules, there are some additional restrictions:

- Do not transfer nonexecutable load modules, or load modules of size 0 or undefined size. Unpredictable results will occur.
- The current working directory on both the client and the server must be the source or destination load library. A load library is a PDS or PDSE with RECFM=U.
- Only member names can be specified. No fully qualified names can be specified.
- File rename is not supported on load module transfer.
- Load modules can be transferred only between the same types of libraries. For example, PDS to PDSE transfer is not allowed.
- If load modules are being sent to or from the z/OS FTP client, the client must be started from one of the following environments:
 - TSO terminal session
 - TSO REXX
 - TSO batch
 - TSO background
 - Unix System services terminal session

- A load module loading from a temporary data set will always be a REPLACE operation, overwriting existing members. LMTR is not performed in STOU mode (the user has toggled SUNIQUE on).
- There is no prompting on mput and mget subcommands. All files that match the mask provided are transferred.

In most cases where load module processing cannot be performed, including failure to abide by the restrictions given above, FTP completes the file transfer using normal processing. Any load modules transferred with normal processing are not executable on the target system.

For the examples shown below, the following assumptions are made:

- The contents of load library USER.LINKLIB are:

| Name | Prompt | Alias-of | Size | TTR | AC | AM | RM |
|----------|--------|----------|----------|--------|----|----|-----|
| EZACDOPN | | | 0000D268 | 00160F | 01 | 31 | ANY |
| EZAFTPLC | | | 000E3758 | 00001B | 01 | 31 | ANY |
| FTP | | EZAFTPLC | 000E3758 | 00001B | 01 | 31 | ANY |
| OPING | | EZACDOPN | 0000D268 | 00160F | 01 | 31 | ANY |

- USER1.TESTLIB is a PDS with RECFM=U.

The following example is a sample session involving a load module transfer with debug/trace on. (For clarity, user input is shown offset to the left and notes are contained within // characters.)

```

220-FTPD1 IBM FTP CS V1R2 at MVS097, 21:16:25 on 2003-01-16.
220 Connection will not timeout.
NAME (9.67.43.61:USER1):
user1
>>> USER user1
331 Send password please.
PASSWORD:

>>> PASS
230 USER1 is logged on. Working directory is "USER1.".
Command:
cd 'user.linklib'
>>> CWD 'user.linklib'
250-The working directory may be a load library
250-The working directory "USER.LINKLIB" is a partitioned data set //1//
Command:
lcd 'user1.testlib'
Local directory might be a load library //1//
Local directory name set to partitioned data set USER1.TESTLIB
Command:
get oping
>>> XLMT PDS 0 oping //2//
250 PDS 53864 - send next command for load module transfer //2//
>>> PORT 9,67,43,65,4,41
200 Port request OK.
>>> RETR oping
125-Transferring load module //2//
125 DCB 32760 32760 //2//

```

```

IEBCOPY  FMID HDZ11D0  SERVICE LEVEL UW90570  DATED 19990520 DFSMS
          1.5.0    MVS SP6.0.8 HBB6608  CPU 9672
USER1    OS390R5  OS390R5  16:20:59 TUE 16 NOV 1999 PARM=''
          STANDARD DD NAMES-  SYSIN      SYSPRINT SYSUT1  SYSUT2  SYSUT3
          SYSUT4
OVERRIDING DD NAMES-  SYS00158 SYS00159 SYS00157 SYS00156 SYSUT3
          SYSUT4
VL GETMAIN REQUESTED 280K TO 1M BYTES.  OBTAINED 1M.

          COPY      OUTDD=SYS00156,INDD=((SYS00157,R))
ORIGINAL PDS  (BEFORE UNLOAD) WAS RECFM=U  BLKSIZE=32760 LRECL=0
          KEYLEN=0 OPTCD=X'20' UCBTYPE=X'3010200F' INDC=X'00'
ALLOCATED 2 CONTIGUOUS BUFFERS EACH 111K BYTES.  WORK AREA HAS 757K
          BYTES AVAILABLE.

COPYING FROM PDSU  INDD=SYS00157 VOL=
          DSN=SYS99320.T162100.RA000.USER1.TEMPXLMT.H01
          TO PDS  OUTDD=SYS00156 VOL=CPDLB1 DSN=USER1.TESTLIB
CONTROL TABLE IS 20 BYTES LONG.  WORK AREA HAS 757K BYTES AVAILABLE.
ALLOCATED SECOND BUFFER OF 745K BYTES.  FIRST BUFFER IS NOW 221K
          BYTES.  WORK AREA HAS 11974 BYTES AVAILABLE.
FOLLOWING MEMBER(S) LOADED FROM INPUT DATA SET REFERENCED BY SYS00157
EZACDPIN HAS BEEN SUCCESSFULLY LOADED //4//
OPING HAS BEEN SUCCESSFULLY LOADED //4//
2 OF 2 MEMBERS LOADED FROM INPUT DATA SET REFERENCED BY SYS00157
THERE ARE 135 UNUSED TRACKS IN OUTPUT DATA SET REFERENCED BY SYS00156
THERE ARE 23 UNUSED DIRECTORY BLOCKS IN OUTPUT DIRECTORY
RELEASED 1016K ADDITIONAL BYTES.
END OF JOB - 0 WAS HIGHEST SEVERITY CODE
250 Transfer completed successfully.
63084 bytes transferred in 0.005 seconds.  Transfer rate 12616.80 Kbytes/sec.
Command:

```

In this next example, the user attempts to transfer a load module and rename the load module. As this is not supported, load module processing will exit and normal processing will take over. The transferred load module will not be executable on the target system.

```

cd 'user.linklib'
>>> CWD 'user.linklib'
250-The working directory may be a load library
250-The working directory "USER.LINKLIB" is a partitioned data set
Command:
lcd 'user1.testlib'
Local directory might be a load library
Local directory name set to partitioned data set USER1.TESTLIB
Command:
get oping ping
Load module transfer does not support load module rename //5//
>>> PORT 9,67,43,66,4,41
200 Port request OK.
>>> RETR oping
125 Sending data set USER.LINKLIB(OPING)
250 Transfer completed successfully.
61984 bytes transferred in 0.190 seconds.  Transfer rate 326.23 Kbytes/sec.
Command:

```

Notes:

1. When a cd or lcd subcommand is performed, the user will be notified if the new current directory is eligible for load module transfer processing. If this message or reply is not seen when changing to a directory, load module transfer will not be attempted.
2. There are additional flows between the client and server for load module transfer.
3. The IEBCOPY system utility is invoked by FTP as part of load module transfer processing. The IEBCOPY messages are written on the client invocation for a client and to syslogd for a server in the following situations:
 - An error exists.
 - No error exists and DEBUG FSC(2) is on.

DUMP 31 can be used to verify that IEBCOPY is invoked when a FTP load module transfer is started.

4. The actual load module and all of its aliases are transferred, even though (in this case only) the alias was specified by the user.
5. When load module transfer processing cannot be performed, the user is warned and the transfer might be completed using normal processing. The data set USER.TESTLIB(PING) is not executable on the client system.

Changing local site defaults using FTP.DATA

The default values for the local site parameters are hardcoded in the FTP client module. You can change these default values by creating an FTP.DATA configuration data set.

Note: Unless otherwise indicated, FTP.DATA data set refers to both the /etc/ftp.data z/OS UNIX file and the MVS data set FTP.DATA.

The FTP.DATA configuration data set is optional. If you specify the -f parameter on the FTP invocation, that parameter is used; otherwise, the FTP client uses one of the search orders shown in [Table 14 on page 62](#) to obtain the local site parameter values.

Table 14. FTP client search orders

| TSO | Batch | z/OS UNIX shell |
|---------------------------|---------------------------|---------------------------|
| 1. SYSFTPD DD statement | 1. SYSFTPD DD statement | 1. \$HOME/ftp.data |
| 2. tso_prefix.FTP.DATA | 2. userid.FTP.DATA | 2. userid.FTP.DATA |
| 3. userid.FTP.DATA | 3. /etc/ftp.data | 3. /etc/ftp.data |
| 4. /etc/ftp.data | 4. SYS1.TCPPARMS(FTPDATA) | 4. SYS1.TCPPARMS(FTPDATA) |
| 5. SYS1.TCPPARMS(FTPDATA) | 5. tcpip_hlq.FTP.DATA | 5. tcpip_hlq.FTP.DATA |
| 6. tcpip_hlq.FTP.DATA | | |

See [z/OS Communications Server: IP Programmer's Guide and Reference](#) for a description of the FTP.DATA search order used when FTP is started from the [FTP Client API](#).

The [z/OS Communications Server: IP Configuration Reference](#) describes the statements you can code in the FTP.DATA data set. It is not necessary to include all statements in the FTP.DATA data set; only those statements whose default values are to be changed must be included. The hard-coded default is used for any statement that is not included in the FTP.DATA data set.

You can find a sample FTP.DATA file in the SEZAINST (FTCDATA) data set.

You can change several of the FTP local site parameters during the FTP session by using the LOCSITE subcommand. See [“LOCSITE subcommand—Specify site information to the local host” on page 179](#) for more information about using the LOCSITE subcommand to change the local site parameters.

Setting user-level options using FTPS.RC

The default values for the site parameters are coded in the server FTP.DATA file. As a client user, you can change the site parameters by creating an FTPS.RC configuration data set. This file can have CD and SITE subcommands.

The FTPS.RC configuration data set is optional. The FTP server uses the following search order to find the data set:

1. tso_prefix.FTPS.RC
2. userid.FTPS.RC
3. \$HOME/ftps.rc

The following are the syntax rules for the FTPS.RC file:

- Only SITE and CWD commands are allowed for this file. (CD is also accepted and is treated as CWD.)

- Each command must be contained in a single line.
- The SITE command can have multiple parameters.
- Comments are allowed, but the entire line will be commented out. In other words, comments cannot exist on the same line as the actual command line.
- Each comment must begin with a semicolon.
- Comments can be entered between two or more command lines.

Note: Test your configuration file to ensure that it is processed correctly by naming the file (following the preceding naming convention) and putting the file into the correct directory. A 230-type reply of Processing FTPS.RC configuration file.. is displayed during your login process. The 230-type reply indicates any errors encountered or whether the file was successfully opened and processed. Contact the system programmer if you cannot fix any errors reported or if the file cannot be opened or created. See [z/OS Communications Server: IP and SNA Codes](#) for additional information about 230-type replies.

The following are examples of an FTPS.RC file with no errors and with errors.

- No errors (correctly coded).

```
; This is a sample configuration file for FTPS.RC
; You may enter comments by starting a line with semicolon (;). The
; entire line will be ignored.
; The server FTP.DATA file must have the statement DEBUGONSITE TRUE
; for the SITE DEBUG= subcommand to be accepted by the server.
SITE debug=all
; You may also insert comments between command lines.
; You may code either 'CD' or 'CWD'.
CWD /user2
CD tmp
```

- Errors within the file.

```
; This is a sample configuration file for FTPS.RC
; You may enter comments by starting a line with semicolon (;). The
; entire line will be ignored.
; The line below has an error: erroremount is an unrecognized parameter.
SITE erroremount
; PUT subcommand is not accepted in this file.
PUT 'user2.tmp.banner' 'user3.tmp.banner'
```

Configuring the FTP client for SOCKS server

The FTP client uses a SOCKS configuration data set or file to determine whether to access a given FTP server directly or through a SOCKS server. The name of the data set or file is specified by coding the SOCKSCONFIGFILE statement in the client FTP.DATA file.

The FTP client references the SOCKSCONFIGFILE only when the FTP server is known to it by an IPv4 IP address or by a DNS name that resolves to an IPv4 IP address. The FTP client always connects directly to FTP servers known to it by IPv6 addresses or by DNS names that resolve to IPv6 addresses.

See [z/OS Communications Server: IP Configuration Reference](#) for details regarding the contents of the SOCKS configuration file.

Sample FTP.DATA data set (FTCDATA)

The following is a sample of the contents of the FTP.DATA data set in the FTCDATA member of the SEZAINST data set.

```

;*****
;
; Name of File:          SEZAINST(FTCDATA)
;
; Descriptive Name:      FTP.DATA  (for FTP Client)
;
; SMP/E Distribution Name:  EZAFTPAC
;
; Copyright:    Licensed Materials - Property of IBM
;
;               "Restricted Materials of IBM"
;
;               5694-A01
;
;               Copyright IBM Corp. 1977, 2011
;
;               US Government Users Restricted Rights -
;               Use, duplication or disclosure restricted by
;               GSA ADP Schedule Contract with IBM Corp.
;
; Status:        CSV1R13
;
; This FTP.DATA file is used to specify default file and disk
; parameters used by the FTP client.
;
; Note: For an example of an FTP.DATA file for the FTP server,
; see the FTPSDATA example.
;
; Syntax Rules for the FTP.DATA Configuration File:
;
; (a) All characters to the right of and including a ; will be
;     treated as a comment.
;
; (b) Blanks and <end-of-line> are used to delimit tokens.
;
; (c) The format for each statement is:
;
;     parameter value
;
; The FTP.DATA options are grouped into the following groups in
; this sample FTP client FTP.DATA configuration data set:
;
; 1. Basic configuration options
; 2. Unix System Services file options
; 3. Default attributes for MVS data set creation
; 4. MVS data set transfer options
; 5. Code page conversion options
; 6. DB2 (SQL) interface options
; 7. Security options
; 8. Timers
; 9. Return codes
; 10. Checkpoint / Restart options
; 11. Socks server access
; 12. Debug (trace) options
; 13. Additional advanced options
;
; For options that have a pre-selected set of values, a (D) indicates*
; the default value for the option.
;
; Options that can be changed via LOCSITE subcommands are identified *
; with an (S).
;
;*****

```

Figure 3. FTP.DATA for FTP client

```

; -----
;
; 1. Basic FTP client configuration options
;
; -----
;SUPPRESSIGNOREWARNINGS FALSE ; Suppress message EZYFT47I
;                               ; while processing remaining
;                               ; statements in this FTP.DATA
;                               ; TRUE - Yes
;                               ; FALSE (D) - No. EZYFT47I is
;                               ; issued to warn of ignored
;                               ; statements
FILETYPE          SEQ          ; (S) Client mode of operation
;                               ; SEQ = transfer data sets or
;                               ; files (D)
;                               ; SQL = submit queries to DB2
;SEQNUMSUPPORT FALSE          ; Support sequence numbers when input
;                               ; read from //INPUT DD file
;                               ; FALSE = (D) Do not support
;                               ; sequence numbers.
;                               ; EZYFS33I issued if
;                               ; sequence numbers detected
;                               ; TRUE = Support sequence numbers

```

```

; -----
;
; 2. Unix System Services file options
;
; -----
UMASK              027          ; (S) Octal UMASK to restrict setting
;                               ; of permission bits when creating
;                               ; new z/OS Unix files and named
;                               ; pipes.
;                               ; Default value is 027.
LISTSUBDIR         TRUE         ; Should wildcard searches span
;                               ; subdirectories?
;                               ; TRUE (D) - Yes
;                               ; FALSE - No
;UNIXFILETYPE      FILE         ; (S) Unix System Services file type
;                               ; FILE (D) - Treat files as regular
;                               ; Unix files
;                               ; FIFO - Treat files as Unix named
;                               ; pipes
;FIFOOPENTIME      60           ; (S) FIFO open timeout in seconds when
;                               ; opening a Unix named pipe.
;                               ; Default value is 60 seconds.
;                               ; Valid range is 1 through 86400.
;FIFOIOTIME        20           ; (S) FIFO timeout for I/O to or from a
;                               ; Unix named pipe
;                               ; Default value is 20 seconds.
;                               ; Valid range is 1 through 86400.

```

```

; -----
;
; 3. Default MVS data set creation attributes
;
; -----
BLKSIZE          6233          ; (S) New data set allocation block
;                               ; size
;                               ; Default is 6233
;                               ; Valid range is 0 to 32760
;DATACLASS        SMSDATA      ; (S) SMS data class name
;                               ; There is no default
;MGMTCLASS        SMSMGNT      ; (S) SMS mgmtclass name
;                               ; There is no default
;STORCLASS        SMSSTOR      ; (S) SMS storclass name
;                               ; There is no default
;DCBDSN          MODEL.DCB     ; (S) New data set allocation
;                               ; model DCB name - must be a
;                               ; fully qualified data set name
;                               ; There is no default
;
; DIRECTORY       27           ; (S) Number of directory blocks in
;                               ; new PDS/PDSE data sets.
;                               ; Default value is 27.
;                               ; Range is from 1 to 16777215.
;DSNTYPE          SYSTEM       ; (S) New data set allocation DSNTYPE
;                               ; for physical sequential data sets
;                               ; BASIC = allocate basic format
;                               ; data set
;                               ; LARGE = allocate large format
;                               ; data set
;                               ; SYSTEM = use system default (D)
;LRECL            256          ; (S) New data set allocation LRECL.
;                               ; Default value is 256.
;                               ; Valid range 0 through 32760.
;PDSTYPE          ; (S) no value - allocate MVS
;                               ; directories according to the
;                               ; system default (PDS or PDSE)
;                               ; PDS - allocate MVS
;                               ; directories as a PDS
;                               ; PDSE - allocate MVS directories
;                               ; as a PDSE
;PRIMARY          1            ; (S) New data set allocation
;                               ; primary space units according
;                               ; to the value of SPACETYPE.
;                               ; Default value is 1.
;                               ; Valid range 1 through 16777215.
;RECFM            VB           ; (S) New data set allocation
;                               ; record format.
;                               ; Default value is VB.
;                               ; Value may be specified as certain
;                               ; combinations of:
;                               ; A - ASA print control
;                               ; B - Blocked
;                               ; F - Fixed length records
;                               ; M - Machine print control
;                               ; S - Spanned (V) or Standard (F)
;                               ; U - Undefined record length
;                               ; V - Variable length records
;RETPD            ; (S) New data set retention
;                               ; period in days.
;                               ; Blank = no retention period (D)
;                               ; 0 = expire today
;                               ; Valid range 0 through 9999.
;                               ; NB: Note the difference between
;                               ; a blank value and a value
;                               ; of zero.

```

| | | |
|-----------|---------------|--|
| SECONDARY | 1 | ; (S) New data set allocation ; secondary space units according ; to the value of SPACETYPE. ; Default value is 1. ; Valid range 1 through 16777215. |
| SPACETYPE | TRACK | ; (S) New data set allocation ; space type. ; TRACK (D) ; BLOCK ; CYLINDER |
| UCOUNT | | ; (S) Sets the unit count for an ; allocation. ; If this option is not specified ; or is specified with a value of ; blank, the unit count attribute ; is not used on an allocation (D) ; Valid range is 1 through 59 or ; the character P for parallel ; mount requests |
| ;UNITNAME | SYSDA | ; (S) New data set allocation unit ; name. ; There is no default. |
| VCOUNT | 59 | ; (S) Volume count for an ; allocation. ; Valid range is 1 through 255. ; Default value is 59. |
| ;VOLUME | WRKLB1,WRKLB2 | ; (S) Volume serial number(s) to ; use for allocating a data set. ; Specify either a single VOLSER ; or a list of VOLSERS ; separated with commas |
| ;EATTR | SYSTEM | ; (S) New data set allocation EATTR ; specifies whether new data sets ; can have extended attributes and ; whether the data sets can reside ; in the EAS. ; NO = no extended attributes ; OPT = yes if volume supports them ; SYSTEM = use system default (D) |

```

; -----
;
; 4. MVS data set transfer options
;
; -----
ASATRANS          FALSE      ; (S) Conversion of ASA print
                        ; control characters
                        ; TRUE = Use C conversion
                        ; FALSE = Do not convert (D)
AUTOMOUNT         TRUE       ; (S) Automatic mount of unmounted
                        ; DASD volumes
                        ; TRUE = Mount volumes (D)
                        ; FALSE = Do not mount volumes
AUTORECALL        TRUE       ; (S) Automatic recall of
                        ; migrated data sets
                        ; TRUE = Recall them (D)
                        ; FALSE = Do not recall them
AUTOTAPEMOUNT     FALSE      ; Automatic mount of unmounted
                        ; tape volumes
                        ; TRUE = Mount volumes
                        ; FALSE = Do not mount volumes (D)
BUFNO             5          ; (S) Specify number of access
                        ; method buffers
                        ; Valid range is from 1 through
                        ; 35 - default value is 5
CONDDISP          CATLG      ; (S) Disposition of a new data set
                        ; when transfer ends prematurely
                        ; CATLG = Keep and catalog (D)
                        ; DELETE = Delete data set
                        ; This option applies to z/OS Unix
                        ; files also
DIRECTORYMODE     FALSE      ; (S) Specifies how to view the MVS
                        ; data set structure:
                        ; FALSE = (D) All qualifiers below
                        ; LCWD are treated as
                        ; entries in the directory
                        ; TRUE = Qualifiers immediately
                        ; below the LCWD are
                        ; treated as entries in the
                        ; directory
ISPFSTATS         FALSE      ; (S) TRUE = create/update PDS
                        ; statistics
                        ; FALSE = (D) does not create /
                        ; update PDS statistics
MIGRATEVOL        MIGRAT     ; (S) Migration volume VOLSER to
                        ; identify migrated data sets
                        ; under control of non-HSM
                        ; storage management products.
                        ; Default value is MIGRAT.
QUOTESOVERRIDE    TRUE       ; (S) How to treat quotes at the
                        ; beginning or surrounding file
                        ; names.
                        ; TRUE = Override current working
                        ; directory (D)
                        ; FALSE = Treat quotes as part of
                        ; file name
RDW               FALSE      ; (S) Specify whether Record
                        ; Descriptor Words (RDWs) are
                        ; discarded or retained.
                        ; TRUE = Retain RDWs and transfer
                        ; as part of data
                        ; FALSE = Discard RDWs when
                        ; transferring data (D)
;READVB          LE          ; (S) Specifies whether variable length
                        ; MVS data sets are read using LE
                        ; or BSAM (low level I/O)
                        ; BSAM = Use BSAM
                        ; LE = Use LE (D)
;

```

| | | |
|----------------|-------|---|
| TRAILINGBLANKS | FALSE | ; (S) How to handle trailing blanks ; in fixed format data sets during ; text transfers. ; TRUE = Retain trailing blanks ; (include in transfer) ; FALSE = Strip off trailing ; blanks (D) |
| TRUNCATE | FALSE | ; (S) Used in conjunction with ; WRAPRECORD to specify what to do ; if no new-line is encountered ; before reaching the MVS data set ; record length limit as defined ; by LRECL when transferring data ; to MVS. This parameter only has ; meaning if WRAPRECORD is false. ; TRUE (D) = allow truncation and ; continue with the file transfer ; FALSE = fail the file ; transfer instead of truncating |
| WRAPRECORD | FALSE | ; (S) Specify what to do if no new-line ; is encountered before reaching ; the MVS data set record length ; limit as defined by LRECL when ; transferring data to MVS. ; TRUE = Wrap data to new record ; FALSE = Truncate data (D) |
| WRTAPEFASTIO | FALSE | ; (S) How should the server write ; ASCII stream mode to tapes? ; TRUE = Use BSAM I/O routines ; FALSE (D) = Use LE Run Time ; library fwrite |

```

; -----
;
; 5. Text code page conversion options
;
; -----
;CCTTRANS          dsn_qual          ; Control connection translate
;                                     ; table data set qualifier.
;                                     ; Used to search for
;                                     ;   a) userid.dsn_qual.TCPXLBIN
;                                     ;   b) hlq.dsn_qual.TCPXLBIN
;                                     ; If CTRLCONN is specified, that
;                                     ; value overrides CCTTRANS.
;CTRLCONN          7BIT              ; (S) ASCII code page for
;                                     ; control connection.
;                                     ; 7BIT is the default if CTRLCONN
;                                     ; is not specified AND no TCPXLBIN
;                                     ; translation table data set found.
;                                     ; Can be specified as any iconv
;                                     ; supported ASCII code page, such
;                                     ; as IBM-850
;DBSUB             FALSE             ; (S) Specifies whether untranslatable
;                                     ; data bytes should be replaced
;                                     ; with substitution character in
;                                     ; iconv() during data transfer.
;                                     ; TRUE = Replace each
;                                     ; untranslatable byte
;                                     ; FALSE = Terminate transfer (D)
;                                     ; when untranslatable bytes are
;                                     ; detected
;ENCODING          SBCS              ; (S) Specifies whether multi-byte or
;                                     ; single-byte data conversion is
;                                     ; to be performed on ASCII data
;                                     ; transfers.
;                                     ; MBCS = Use multi-byte
;                                     ; SBCS = Use single-byte (D)
;EXTDBSCHINESE     TRUE              ; (S) Specifies whether to use extended
;                                     ; double byte range for Simplified
;                                     ; Chinese or the old range.
;                                     ; TRUE = (D) Use the extended range
;                                     ; 1st byte x'81' - x'FE'
;                                     ; 2nd byte x'40' - x'FE'
;                                     ; FALSE= Use the range of
;                                     ; 1st byte x'8C' - x'FE'
;                                     ; 2nd byte x'A1' - x'FE'
;EXTENSIONS        UTF8              ; Enable RFC 2640 support.
;                                     ; Default is disabled.
;                                     ; Control connection starts as
;                                     ; 7bit ASCII and switches to UTF-8
;                                     ; encoding when LANG command
;                                     ; processed successfully. CCTTRANS
;                                     ; and CTRLCONN are ignored.
;MBDATACONN        (IBM-1388,IBM-5488) ; (S) Specifies the conversion table
;                                     ; names for the data connection
;                                     ; when ENCODING has a value of
;                                     ; MBCS. The names are the file
;                                     ; system code page name and the
;                                     ; network transfer code page name.

```



```

;MSENDEOL          CRLF          ; (S) When translating multi-byte data
;                               ; to ASCII :
;                               ; CRLF = (D) Append a carriage
;                               ; return (x'0D') and line
;                               ; feed (x'0A') to each line
;                               ; of text. This is the
;                               ; default and the standard
;                               ; line terminator defined by
;                               ; RFC 959. The z/OS server
;                               ; and client can receive
;                               ; ASCII data only in this
;                               ; format.
;                               ; CR   = Append a carriage return
;                               ; (x'0D') only to each line
;                               ; of text.
;                               ; LF   = Append a line feed (x'0A')
;                               ; only to each line of text.
;                               ; NONE = Do not append a line
;                               ; terminator to any line of
;                               ; text.
;MBREQUIRELASTEOL TRUE          ; (S) Specifies whether the last
;                               ; record of an incoming multibyte
;                               ; transfer is required to have
;                               ; an EOL sequence.
;                               ; TRUE  A missing EOL on the last
;                               ; record received is treated as an
;                               ; error (D)
;                               ; FALSE A missing EOL on the last
;                               ; record received is ignored
;REMOVEINBEOF      FALSE          ; (S) Remove final UNIX EOF from
;                               ; inbound ASCII transfers
;                               ; TRUE - final UNIX EOF is removed
;                               ; FALSE - final UNIX EOF is not
;                               ; removed (D)
;SBDATACONN (IBM-1047,IBM-850) ; (S) file system/network transfer
;                               ; code pages for data connection.
;                               ; Either a fully-qualified MVS
;                               ; data set name or z/OS Unix file
;                               ; name built with the CONVXLAT ;
;                               ; utility -
;                               ; HLQ.MY.TRANS.DATASET
;                               ; /u/user1/my.trans.file
;                               ; Or a file system code page name
;                               ; followed by a network transfer
;                               ; code page name according to
;                               ; iconv supported code pages -
;                               ; for example
;                               ; (IBM-1047,IBM-850)
;                               ; If SBDATACONN is not present,
;                               ; std. search order for a default
;                               ; translation table data set will
;                               ; be used.

```

```

;SBSENDEOL      CRLF      ; (S) When translating single-byte
;                  ; data to ASCII :
;                  ; CRLF = (D) Append a carriage
;                  ;      return (x'0D') and line
;                  ;      feed (x'0A') to each line
;                  ;      of text. This is the
;                  ;      default and the standard
;                  ;      line terminator defined by
;                  ;      RFC 959. The z/OS server
;                  ;      and client can receive
;                  ;      ASCII data only in this
;                  ;      format.
;                  ; CR   = Append a carriage return
;                  ;      (x'0D') only to each line
;                  ;      of text.
;                  ; LF   = Append a line feed (x'0A')
;                  ;      only to each line of text.
;                  ; NONE = Do not append a line
;                  ;      terminator to any line of
;                  ;      text.
;SBSUB           FALSE     ; (S) Specifies whether untranslatable
;                  ; data bytes should be replaced
;                  ; with SBSUBCHAR when detected
;                  ; during SBCS data transfer.
;                  ; TRUE  = Replace each
;                  ;      untranslatable byte with
;                  ;      SBSUBCHAR.
;                  ; FALSE = Terminate transfer (D)
;                  ;      when untranslatable bytes are
;                  ;      detected
;SBSUBCHAR        SPACE     ; (S) Specifies the substitution char
;                  ; for SBCS data transfer when
;                  ; SBSUB is TRUE.
;                  ; nn    = hexadecimal value from
;                  ;      0x'00' to 0xFF'.
;                  ; SPACE = x'40' when target code
;                  ;      set is EBCDIC, and
;                  ;      x'20' when target code
;                  ;      set is ASCII. (D)
;SBTRANS          dsn_qual  ; Data connection translate
;                  ; table data set qualifier.
;                  ; Used to search for
;                  ; a) userid.dsn_qual.TCPXLBIN
;                  ; b) hlq.dsn_qual.TCPXLBIN
;                  ; If SBDATACONN is specified, that
;                  ; value overrides SBTRANS

```

```

;UCSHOSTCS      code_set      ; (S) Specify the EBCDIC code set
;                                     ; to be used for data conversion
;                                     ; to or from UCS-2.
;                                     ; If UCSHOSTCS is not specified,
;                                     ; the current EBCDIC code page
;                                     ; for the data connection is used.
UCSSUB          FALSE         ; (S) Specify whether UCS-2 to EBCDIC
;                                     ; conversion should use the EBCDIC
;                                     ; substitution character or
;                                     ; cause the data transfer to be
;                                     ; terminated if a UCS-2 character
;                                     ; cannot be converted to a
;                                     ; character in the target EBCDIC
;                                     ; code set
;                                     ; TRUE = Use substitution char
;                                     ; FALSE = Terminate transfer (D)
UCSTRUNC        FALSE         ; (S) Specify whether the transfer
;                                     ; of UCS-2 data should be
;                                     ; aborted if truncation
;                                     ; occurs at the MVS host
;                                     ; TRUE = Truncation allowed
;                                     ; FALSE = Terminate transfer (D)
;UNICODEFILESYSTEMBOM ASIS    ; (S) When storing UNICODE files,
;                                     ; specifies whether to store a
;                                     ; Byte Order Mark (BOM) as the
;                                     ; first character of the file.
;                                     ; ASIS = (D) Store a BOM if one
;                                     ; was transmitted with the file
;                                     ; as the first character.
;                                     ; ALWAYS = Always store a BOM as
;                                     ; the first character of the file
;                                     ; NEVER = Never store a BOM as
;                                     ; the first character of the file
;                                     ; regardless of whether a BOM was
;                                     ; was sent. Although a BOM can
;                                     ; appear anywhere within the
;                                     ; file, only a BOM sent as the
;                                     ; first file character is
;                                     ; affected by this setting.

```

```

; -----
;
; 6. DB2 (SQL) interface options
;
; -----
DB2              DB2          ; (S) DB2 subsystem name
;                                     ; The default name is DB2
DB2PLAN          EZAFTPMQ     ; DB2 plan name for FTP client
;                                     ; The default name is EZAFTPMQ
SPREAD          FALSE         ; (S) SQL spreadsheet output format
;                                     ; TRUE = Spreadsheet format
;                                     ; FALSE = Not spreadsheet
;                                     ; format (D)
SQLCOL          NAMES         ; (S) SQL output headings
;                                     ; NAMES = Use column names (D)
;                                     ; LABELS = Use column labels
;                                     ; ANY = Use label if defined,
;                                     ; else use name

```

```

; -----
; 7. Security options
; -----
;SECURE_MECHANISM  GSSAPI          ; Name of the security mechanism
;                                     ; that the client uses when it
;                                     ; sends an AUTH command to the
;                                     ; server.
;                                     ; GSSAPI = Kerberos support
;                                     ; TLS      = TLS
;SECURE_FTP         ALLOWED         ; Authentication indicator
;                                     ; ALLOWED      (D)
;                                     ; REQUIRED
;SECURE_CTRLCONN    CLEAR          ; Minimum level of security for
;                                     ; the control connection
;                                     ; CLEAR        (D)
;                                     ; SAFE
;                                     ; PRIVATE
;SECURE_DATACONN    CLEAR          ; Minimum level of security for
;                                     ; the data connection
;                                     ; NEVER
;                                     ; CLEAR        (D)
;                                     ; SAFE
;                                     ; PRIVATE
;SECURE_HOSTNAME    OPTIONAL       ; Authentication of hostname in
;                                     ; the server certificate
;                                     ; OPTIONAL (D)
;                                     ; REQUIRED
;SECURE_PBSZ        16384          ; Kerberos maximum size of the
;                                     ; encoded data blocks
;                                     ; Default value is 16384
;                                     ; Valid range is 512 through 32768
; Name of a ciphersuite that can be passed to the partner during
; the TLS handshake. None, some, or all of the following may be
; specified. The number to the far right is the cipherspec id
; that corresponds to the ciphersuite's name.
;CIPHERSUITE        SSL_NULL_MD5    ; 01
;CIPHERSUITE        SSL_NULL_SHA    ; 02
;CIPHERSUITE        SSL_RC4_MD5_EX  ; 03
;CIPHERSUITE        SSL_RC4_MD5     ; 04
;CIPHERSUITE        SSL_RC4_SHA     ; 05
;CIPHERSUITE        SSL_RC2_MD5_EX  ; 06
;CIPHERSUITE        SSL_DES_SHA     ; 09
;CIPHERSUITE        SSL_3DES_SHA    ; 0A
;CIPHERSUITE        SSL_AES_128_SHA ; 2F
;CIPHERSUITE        SSL_AES_256_SHA ; 35
;KEYRING            name           ; Name of the keyring for TLS
;                                     ; It can be the name of a z/OS Unix
;                                     ; file (name starts with /) or
;                                     ; a resource name in the security
;                                     ; product (e.g., RACF)
;TLSTIMEOUT         100           ; Maximum time limit between full
;                                     ; TLS handshakes to protect data
;                                     ; connections
;                                     ; Default value is 100 seconds.
;                                     ; Valid range is 0 through 86400

;SECUREIMPLICITZOS  TRUE          ; (S) Specify whether client will
;                                     ; connect to a z/OS FTP server
;                                     ; when connecting to the TLS port.
;                                     ; TRUE (D)
;                                     ; FALSE Use FALSE if server is
;                                     ; not z/OS or when not connecting
;                                     ; to the TLS port as specified by
;                                     ; the TLSPORT statement.
;TLSPORT            990           ; Specify which FTP port is
;                                     ; implicitly secured with TLS
;                                     ; 0 disable implicit security
;                                     ; 990 (D) default value
;                                     ; Valid range is 0 to 65534
;TLSRFCLEVEL        DRAFT         ; (S) Specify what level of RFC 4217,
;                                     ; On Securing ; FTP with TLS, is
;                                     ; supported
;                                     ; DRAFT (D) Internet Draft level
;                                     ; RFC4217 RFC level

```

```

; -----
; 8. Timers
; -----
CCONNTIME      30      ; Timeout value for successful
; close of control connection.
; Default value is 30 seconds.
; Valid range is 15 through 86400.
; 0 = do not timeout
DATACTIME      120     ; Timeout for send/receive data
; operations.
; Default value is 120 seconds.
; Valid range is 15 through 86400.
; 0 = do not timeout
;DATAKEEPAIVE  0      ; (S) Keepalive packets are sent
; after the data connection is
; idle for the specified number
; of seconds on the data
; connection.
; 0 seconds (D)
; 0 = use keepalive interval
; configured in the PROFILE.TCPIP
; for passive mode and no keepalive
; packets for active mode
; Valid range is 60 - 86400
DCONNTIME      120     ; Timeout value for successful
; close of data connection.
; Default value is 120 seconds.
; Valid range is 15 through 86400.
; 0 = do not timeout
;DSWAITTIME    0      ; (S) The approximate number of
; minutes ftp waits when trying
; to access an MVS data set.
; Default is 0 minutes
; 0 (D)
; Valid range is 0 - 14400
FTPKEEPAIVE     0      ; Keepalive packets are sent after
; the control connection is
; idle for the specified number
; of seconds
; Default is 0 seconds
; 0 = do not send keepalive packets
; Valid range is 60 - 86400
INACTTIME      120     ; The time in seconds to wait for
; an expected response from the
; server.
; Default value is 120 seconds.
; Valid range is 15 through 86400.
; 0 = do not timeout
MYOPENTIME     60      ; Connection timeout value in
; seconds.
; Default value is 60 seconds.
; Valid range is 15 through 86400.
; 0 = do not timeout
PROGRESS       10      ; Time interval in seconds between
; progress updates for file
; transfers. Default is 10 seconds
; Valid range is 10 through 86400,
; or 0 to request no updates.

```

```

; -----
; 9. Return codes
; -----
;CLIENTERRCODES  FALSE ; Return code format
; TRUE - 2 digit error return code
; FALSE (D) - 5 digit XYYYY format
;   XX - FTP subcommand
;   YYY - server reply code
; EXTENDED - 4 digit XYYY format
;   XX - 2 digit error return code
;   YY - FTP subcommand
;LOGCLIENTERR    FALSE ; Report errors with EZZ9830I msg?
; TRUE          - Yes
; FALSE (D)     - No

```

```

; -----
;
; 10. Checkpoint / Restart options
; -----
CHKPTINT          0          ; (S) Specify the checkpoint interval
                        ; in number of records.
                        ; NB: checkpointing only works
                        ; with datatype EBCDIC and block
                        ; or compressed transfer mode.
                        ; 0 = no checkpoints (D)
RESTGET           TRUE       ; (S) Should checkpointing occur during
                        ; a GET operation?
                        ; TRUE (D) - Yes
                        ; FALSE - No
CHKPTPREFIX       HOME       ; (S) Low level qualifier of checkpoint
                        ; data set: FTP.CHECKPOINT
                        ; HOME (D) - either TSO prefix or
                        ; UNIX local directory path
                        ; USERID - login user ID
                        ; LOCAL - current local directory

```

```

; -----
; 11. SOCKS server options
; -----
;
; SOCKSCONFIGFILE  /etc/socks.conf ; file path for SOCKS configuration
                        ; file. The SOCKS configuration
                        ; file specifies which FTP servers
                        ; should be accessed via SOCKS.

```

```

; -----
; 12. Debug (trace) options
; -----
;
; DEBUG           TIME       ; time stamp client trace entries
; DEBUG           ALL        ; activate all traces
; DEBUG           BAS        ; active basic traces
                        ; (marked with an *)
; DEBUG           FLO        ; function flow
; DEBUG           CMD        ; * command trace
; DEBUG           PAR        ; parser details
; DEBUG           INT        ; * program initialization and
                        ; termination
; DEBUG           ACC        ; access control (logging in)
; DEBUG           SEC        ; security processing
; DEBUG           UTL        ; utility functions
; DEBUG           FSC(1)     ; * file services
; DEBUG           SOC(1)     ; * socket services
; DEBUG           SQL        ; special SQL processing

```

```

; -----
;
; 13. Additional advanced options
; -----
CHKCONFIDENCE      FALSE      ; (S) FALSE = (D) Do not perform
;                               ; confidence checks of
;                               ; data transfers.
;                               ; TRUE  = Check and report on
;                               ; the confidence in the
;                               ; successful completion of
;                               ; a data transfer. The FTP
;                               ; client reports the level
;                               ; of confidence after each
;                               ; file transfer with the
;                               ; message EZA2108I.
;FWFRIENDLY         FALSE      ; (S) Use firewall friendly protocol
;                               ; for starting data connections?
;                               ; TRUE - Yes
;                               ; FALSE (D) - NO
;EPSV4              FALSE      ; (S) Use NAT firewall friendly protoco
;                               ; for starting data connections?
;                               ; TRUE - Yes
;                               ; FALSE (D) - NO
;PASSIVEIGNOREADDR  FALSE      ; (S) Specifies whether the FTP client
;                               ; should ignore the IP address in
;                               ; the FTP server PASV reply for
;                               ; the data connection and use the
;                               ; IP address that was used to log
;                               ; into the FTP server.
;                               ; TRUE - Ignore FTP Server PASV
;                               ; reply IP address
;                               ; FALSE (D) - Use FTP Server PASV
;                               ; reply IP address
;NETRCLEVEL         1          ; When logging in, should the FTP
;                               ; server's IP addr be converted to
;                               ; a host name to use NETRC login
;                               ; file?
;                               ; 1 (D) - IP addr is not converted
;                               ; 2      - IP addr is converted
;                               ; 2      - IP addr is converted
;TRACECAPI          CONDITIONAL ; When the FTP client is invoked
;                               ; from the FTP Callable API, write
;                               ; records to the API trace spool
;                               ; data set based on this setting
;                               ; CONDITIONAL (D)
;                               ; Trace requests for which the
;                               ; application has set the
;                               ; FCAI_TraceIt field to
;                               ; FCAI_TraceIt_Yes (1)
;                               ; ALL
;                               ; Trace all requests, regardless
;                               ; of the value in FCAI_TraceIt
;                               ; NONE
;                               ; Trace no requests, regardless
;                               ; of the value in FCAI_TraceIt

```

FTP data conversion

By default, the z/OS FTP client transmits all data on the control connection and on the data connection as single byte ASCII, the same ASCII code page being used to encode both connections. z/OS FTP provides the ability to specify different code pages for the control and data connections. The code page you specify can be single byte (SBCS), double byte (DBCS), or multibyte (MBCS).

Support for SBCS languages

SBCS (single byte character set) encoding is the default encoding for both control and data connections, and is the encoding specified in FTP RFCs such as RFC 959. You can specify SBCS encoding on either the control or data connection.

Some methods of specifying alternate translation tables for the FTP client apply to both the control and data connection. If the translation table you need for data transfer does not support the standard

encodings for the portable character set, you should establish different translation tables for the control and data connections to ensure that FTP commands and replies are translated correctly.

Specify SBCS encoding for the data connection with one of the following methods:

- Code statements in FTP.DATA: SBADATACONN and SBTRANS.
- Use LOCSITE SBDataconn or LOCSITE XLATE subcommands to set the code page.
- Place an MVS data set containing a binary translate table in the FTP client's search order for TCPXLBIN data sets. Use the CONVXLAT utility to generate an MVS data set containing the binary translation table you require. See [SBCS translation table hierarchy in the z/OS Communications Server: IP Configuration Reference](#) for the search order used by the FTP client.
- Specify the TRANSLATE parameter as an FTP client start option. The translate parameter applies to both the control and data connections.

If you need to establish different translate tables for the control and data connection, use one of the other methods to establish the translate table, or else change either translate table after starting the client.

If the table you specified with the TRANSLATE parameter does not support the POSIX portable character set, start the client without specifying a host name so the client does not attempt to send commands on the control connection before the correct translation table is established. Then change the client's translation table after starting FTP.

You can specify different conversions for the control connection by using any of the following methods:

- Code CTRLConn statements in the FTP.DATA file.
- Place an MVS data set containing a binary translate table in the FTP client's search order for TCPXLBIN data sets. Use the CONVXLAT utility to generate an MVS data set that contains the binary translation table you require. See [SBCS translation table hierarchy in the z/OS Communications Server: IP Configuration Reference](#) for the search order used by the FTP client.
- Use the LOCSITE CTRLCONN subcommand.
- Specify the TRANSLATE parameter as an FTP client start option. If you use this method, the table applies to both control and data connections.

If you code EXTENSIONS UTF8 in the client FTP.DATA file, the control connection uses 7-bit ASCII for commands and, when negotiated with the server, UTF-8 encoding of path names. The client can override the EXTENSIONS UTF8 statement by using the FTP TRANSLATE start parameter or by issuing SITE and LOCSITE subcommands. However, the client cannot resume UTF-8 encoding on the control connection until you restart the client.

Extended trace point ID 81 is available for tracing the translate tables. When set to ON, 256 bytes of each translate table can be traced as follows:

- When the FTP STAT command is sent to the server, the translate tables being used by the server for the control and data connection are traced. When the FTP LOCSTAT subcommand is entered, the translate tables being used by the client are traced.
- When the LOCSITE subcommand is entered to change the client translate table, the client traces the new table. When the server receives a SITE command to change the translate table, the server traces the new table.

See the [SITE subcommand - DUMP parameter](#) for instructions for activating extended trace point 81 for the server. See the [“DUMP subcommand—Set extended trace options” on page 163](#) for instructions for activating extended trace point 81 for the client.

FTP with traditional DBCS support

This section describes how to use FTP to exchange DBCS data sets between hosts supporting DBCS file transfer.

The z/OS FTP server and client programs access data sets containing data that is usually in EBCDIC format. To transfer these data sets to or from an ASCII-based host requires translation tables. The transfer of DBCS data uses two tables—one for DBCS characters and one for SBCS characters.

Selecting a DBCS translation table

The LOADDBCSTABLES statement in TCPIP.DATA is used by both the FTP server and client to determine which DBCS translation table data sets can be loaded. See [Using translation tables in z/OS Communications Server: IP Configuration Reference](#) for more information about the loading and customizing of DBCS translation tables for FTP.

The FTP server and client can be configured to load a number of DBCS translation tables. These are used during data set transfers to convert MVS host DBCS characters and non-MVS DBCS characters. The FTP command TYPE B n or the corresponding client subcommand is used to enter DBCS transfer mode and select a DBCS table.

Selecting an SBCS translation table

The SBCS table used to transfer DBCS data is the SBCS table that is established for the data connection.

SBCS tables are used by the control connection to transfer commands; they are also used by the data connection. Often the same SBCS table is used, but you might want to select a different table to be used for data transfers. How you specify the SBCS table for the data connection depends on whether the translation is to be done by the FTP server or the FTP client.

When the EBCDIC-to-ASCII translation is done by the FTP server, you can issue a SITE SBDataconn command to select the SBCS table to be used by the server for data transfers.

When the EBCDIC-to-ASCII translation is done by the FTP client, you can use the following parameters in your local FTP.DATA file to establish the SBCS tables:

CTRLConn

Establishes the SBCS tables the client uses for control connections.

SBDataconn

Establishes the SBCS tables the client uses for data connections.

Alternatively, you can use the TRANSLATE option of the FTP command to change the SBCS and DBCS translation table hierarchy for both the control and data connection. The TRANSLATE option results in the same SBCS table for both the control and the data connection.

Note: The TRANSLATE option can be used as long as the table maintains the integrity of the portable character set.

Another alternative when the FTP client is to perform the translation is to use the LOCSITE SBDataconn subcommand to change the SBCS table used by the client for the data connection.

DBCS subcommands

DBCS data sets are transferred using the standard FTP subcommands PUT and GET. However, before the transfer commences, the current transfer type for the session must be set to the required DBCS type. To set the transfer type to DBCS for an FTP session, you must issue the appropriate FTP subcommand to the client or the server, depending on where the DBCS conversion is to be done. The FTP subcommands for DBCS support are listed in [Table 15 on page 79](#).

Table 15. FTP subcommands for DBCS support

| Subcommand | Description | See |
|------------|------------------------------------|---|
| BIG5 | Sets the transfer type to BIG-5 | “BIG5 subcommand—Change the data transfer type to BIG5 ” on page 146 |
| EUckanji | Sets the transfer type to EUCKANJI | “EUckanji subcommand—Change the data transfer type to EUCKANJI” on page 165 |

Table 15. FTP subcommands for DBCS support (continued)

| Subcommand | Description | See |
|------------|---------------------------------------|---|
| HAngeul | Sets the transfer type to HANGEUL | “HAngeul subcommand—Change the data transfer type to HANGEUL ” on page 170 |
| Ibmkanji | Sets the transfer type to IBMKANJI | “Ibmkanji subcommand—Change the data transfer type to IBMKANJI” on page 171 |
| JIS78kj | Sets the transfer type to JIS78KJ | “JIS78kj subcommand—Change the data transfer type to JIS78KJ ” on page 172 |
| JIS83kj | Sets the transfer type to JIS83KJ | “JIS83kj subcommand—Change the data transfer type to JIS83KJ ” on page 173 |
| Ksc5601 | Sets the transfer type to KSC5601 | “Ksc5601 subcommand—Change the data transfer type to KSC-5601 ” on page 173 |
| QUOte | Sends an uninterpreted string of data | “QUOte subcommand—Send an uninterpreted string of data ” on page 240 |
| SChinese | Sets the transfer type to SCHINESE | “SChinese subcommand—Change the data transfer type to SCHINESE” on page 244 |
| SJiskanji | Sets the transfer type to SJISKANJI | “SJiskanji subcommand—Change the data transfer type to SJISKANJI” on page 275 |
| TChinese | Sets the transfer type to TCHINESE | “TChinese subcommand—Change the data transfer type to TCHINESE” on page 287 |
| TyPe | Specifies the transfer type | “TyPe subcommand—Set the data transfer type ” on page 289 |

Server commands and client subcommands

Table 16 on page 80 shows examples of the server command that would be generated for each client subcommand:

Table 16. FTP TYPE subcommand aliases

| Client subcommand | Server command | Description |
|-------------------------|----------------|---|
| BIG5 | TYPE B 8 | Big-5 transfer type |
| EUCKANJI | TYPE B 2 | Extended UNIX Code kanji transfer type |
| HANGEUL | TYPE B 5 | Hangeul transfer type |
| IBMKANJI | TYPE F 1 | IBM (EBCDIC) kanji transfer type |
| JIS78KJ | TYPE B 4 A | JIS 1978 kanji using ASCII shift-in transfer type |
| JIS78KJ (ASCII) | TYPE B 4 A | ASCII shift-in escape sequence |
| JIS78KJ (JISROMAN) | TYPE B 4 R | JISROMAN shift-in escape sequence |
| JIS78KJ (JISROMAN NOSO) | TYPE B 4 R N | Pure DBCS data transfer |
| JIS83KJ | TYPE B 3 A | JIS 1983 kanji using ASCII shift-in transfer type |
| JIS83KJ (ASCII) | TYPE B 3 A | ASCII shift-in escape sequence |
| JIS83KJ (JISROMAN) | TYPE B 3 R | JISROMAN shift-in escape sequence |

Table 16. FTP TYPE subcommand aliases (continued)

| Client subcommand | Server command | Description |
|------------------------|----------------|---|
| JIS83KJ (JISROMAN NOSO | TYPE B 3 R N | Pure DBCS data transfer |
| KSC5601 | TYPE B 6 | Korean Standard Code KSC-5601 transfer type |
| SCHINESE | TYPE B 9 | Simplified Chinese transfer type |
| SJISKANJI | TYPE B 1 | Shift JIS kanji transfer type |
| SJISKANJI (Sosi | TYPE B 1 S A | Shift-out/shift-in characters X'1E'/X'1F' |
| SJISKANJI (Sosi ASCII | TYPE B 1 S A | Shift-out/shift-in characters X'1E'/X'1F' |
| SJISKANJI (Sosi EBCDIC | TYPE B 1 S E | Shift-out/shift-in characters X'0E'/X'0F' |
| SJISKANJI (Sosi SPACE | TYPE B 1 S S | Shift-out/shift-in characters X'20'/X'20' |
| SJISKANJI (NOSO | TYPE B 1 N | Pure DBCS data transfer |
| TCHINESE | TYPE B 7 | Traditional Chinese (5550) transfer type |

Mapping DBCS aliases to CCSIDs

The code sets supported by the DBCS for FTP options conform to standard coded character set identifiers (CCSIDs). [Table 17 on page 81](#) shows how CCSIDs map to DBCS keywords.

For more information about CCSIDs, see *z/OS DFSMS*.

Table 17. Mapping of DBCS keywords to CCSIDs

| DBCS keyword | CCSID | Description |
|--------------|-------|-----------------------------------|
| BIG5 | 00947 | IBM Big-5 DBCS |
| EUCKANJI | 00954 | Japanese EUC (G0, G1 and G2 only) |
| HANGEUL | 00926 | Korean DBCS-PC |
| JIS78KJ | 00955 | JIS X0208–1978 |
| JIS83KJ | 05048 | JIS X0208–1990 |
| KSC5601 | 00951 | IBM Korean Standard code |
| SCHINESE | 01380 | Simplified Chinese DBCS-PC |
| SJISKANJI | 00301 | Japanese DBCS-PC |
| TCHINESE | 00927 | Traditional Chinese DBCS-PC |

Support for MBCS languages

MBCS translation can be performed on the data connection.

The FTP client and server provide double-byte language support using a set of subcommands at the client and corresponding TYPE B commands at the server. This support is described in “FTP with traditional DBCS support” on page 78. An alternative to using the subcommands and TYPE B commands is to use the multibyte support in FTP that is activated by the ENCODING keyword in the FTP.DATA file, LOCSITE subcommand, or SITE command. Use the ENCODING and MBDATACONN keywords to enable translation using system supplied codepages. This method supports most of the double-byte languages currently handled by the traditional DBCS (TYPE B) support.

For example, you can specify that you want to use the Chinese standard GB18030 provided by the codepage IBM-5488 for data conversion on your data connections. To use the codepage IBM-5488, you must specify that multibyte encoding is to be used. You can do this in one of the following ways:

- Code the following statement in the FTP.DATA file:

```
ENCODING MBCS
```

- Issue a subcommand:

```
LOCSITE ENCODING=MBCS
```

You can then specify which codepage the IBM-5488 encoded data is to be converted to or from in the file system by doing one of the following:

- Code one of the following statements in the FTP.DATA file:

```
MBDATACONN (IBM-1388, IBM-5488)
```

or

```
MBDATACONN (UTF-8, IBM-5488)
```

- Issue one of the following subcommands:

```
LOCSITE MBDATACONN=(IBM-1388, IBM-5488)
```

or

```
LOCSITE MBDATACONN=(UTF-8, IBM-5488)
```

Guideline: These steps control the client end of the data connection. To request the same conversions at the server end of the data connection, the same statements must be added to the server FTP.DATA file or use the SITE subcommand.

Specifying values for new data sets

When allocating new data sets, there are several methods you can use to specify the data set attributes. You can individually use the storage attribute parameters with the SITE and LOCSITE subcommands or the *hlq*.FTP.DATA data set. Or, if your system administrator has used the Storage Management Subsystem to group together default attributes into named classes, you can specify those class names on the DATAClass, STORclass, and MGMTclass parameters.

Dynamic allocation of new data sets

FTP enables you to dynamically allocate a new physical-sequential data set, partitioned data set (PDS), or partitioned data set extended (PDSE) for the purpose of transferring data to be written to that data set. The following optional allocation variables can be used by the client to override and turn off the hard-coded defaults that affect the allocation of the data set.

| Variable | FTP.DATA parameter |
|---------------------|--------------------|
| allocation units | SPACETYPE |
| blocksize | BLKSIZE |
| data class | DATAClass |
| directory blocks | DIRECTORY |
| data set name type | DSNTYPE |
| extended attributes | EATTR |

| Variable | FTP.DATA parameter |
|-----------------------|--------------------|
| logical record length | LRECL |
| management class | MGMTCLASS |
| model DCB values | DCBDSN |
| PDS type | PDSTYPE |
| primary space | PRIMARY |
| record format | RECFM |
| retention period | RETPD |
| secondary space | SECONDARY |
| storage class | STORCLASS |
| unit | UNITNAME |
| volume count | VCOUNT |
| unit count | UCOUNT |
| volume serial number | VOLUME |

The MVSGet and MVSPut subcommands affect the values that are configured with the PDSTYPE, DIRECTORY, DSNTYPE, RECFM BLKSIZE, LRECL, PRIMARY, SECONDARY, EATTR and SPACETYPE parameters. The subcommands reset these configured values to match the attributes of the source data set in the same way as if you had configured the values with the SITE or LOCSITE subcommand.

Some of these allocation variables might provide duplicate information. For example, the model DCB might have a record format (RECFm) that differs from the record format specified by a data class and from the one explicitly specified by the client. FTP passes all variables that are specified to dynamic allocation and lets it determine which of the specifications takes precedence. The following list describes the exceptions to that policy:

- If neither the primary nor secondary space quantity is specified, the allocation units value is not sent.
- If the data set organization is physical-sequential, directory blocks specification is not sent.
- If the data set organization is PO (PDS or PDSE), the data set name type specification is not sent.
- Otherwise, all variables are sent to dynamic allocation where the order of precedence is:
 1. Any attributes set by the MVSGet or MVSPut subcommand
 2. Any FTP.DATA, SITE, or LOCSITE configuration options explicitly specified or defaulted
 3. Any attributes picked up from the model DCB and not otherwise explicitly specified
 4. Any attributes picked up from the data class and not previously derived from 1 or 2
 5. Any allocation defaults

Automatically generated SITE subcommand

The FTP client automatically generates a SITE subcommand when sending an MVS data set with the PUT or MPUT subcommand. The SITE subcommand includes information on the record format, logical record length, and block size of the data set. An example of a generated SITE subcommand is:

```
SITE FIXrecfm 80 LRECL=80 RECFM=FB BLKSIZE=320
```

Where: FIXrecfm 80 is intended for use by VM servers and LRECL=80 RECFM=FB BLKSIZE=320 is intended for use by MVS servers. ASCII servers ignore this SITE information. The SENDSITE subcommand can be used to toggle the automatic sending of the SITE subcommand information. See [“SENDSITE subcommand—Toggle the sending of site information”](#) on page 246 for more information.

The MVSPut subcommand automatically generates the SITE subcommand. The MVSPut subcommand affects server PDSTYPE, DIRECTORY, DSNTYPE, RECFM BLKSIZE, LRECL, PRIMARY, SECONDARY, EATTR and SPACETYPE configured values. However, the SITE subcommand is issued regardless of whether the SENDSITE subcommand is issued.

Storage Management Subsystem (SMS)

An FTP client can specify one or more of the Storage Management Subsystem (SMS) classes to manage characteristics that are associated with or assigned to data sets.

- Data class is an SMS construct that determines data set allocation attributes used by SMS for creation of data sets. The fields listed are available attributes that serve as a template for allocation. Each is optional and is overridden by any explicit specification of FTP allocation variables or by a model DCB (DCBDSN).

| Variable | FTP.DATA parameter |
|-----------------------|--------------------|
| directory blocks | DIRECTORY |
| extended attributes | EATTR |
| logical record length | LRECL |
| primary space | PRIMARY |
| record format | RECFM |
| retention period | RETPD |
| secondary space | SECondary |

Note: If either primary or secondary space is explicitly specified, the primary and secondary values from data class are not used.

- Management class is an SMS construct that determines Data Facility Hierarchical Storage Manager (DFHSM) action for data set retention, migration, backup, and release of allocated but unused space. Management class replaces and expands attributes that otherwise would be specified. That is, management class might override any other specification of retention period.
- Storage class is a list of storage performance and availability services requests for an SMS-managed data set that SMS attempts to honor when selecting a volume or volumes for the data set. It might conflict with an explicit specification of volume and unit. If storage class is used, volume and unit parameters should be unspecified.

Steps for using a DCBDSN model to create a new data set

We can use a DCBDSN model to specify the data set attributes to create a new data set.

Procedure

Perform the following steps to use a DCBDSN model to create a data set.

1. Issue the following command:

```
SITE DCBDSN=data_set_name
```

where *data_set_name* is the name of the data set to be used as a model to set the values of the logical record length (LREcl), the block size (BLKSize), the retention period (RETPd), and the record format (RECFm) of a new data set.

2. Issue the following command to enable the LREcl, BLKSize, and RECFm of the model to be used:

```
SITE LRECL BLKSIZE RETPD RECFM
```

-
3. Issue the following command to create the new data set with the values specified by the DCBDSN model:

```
PUT data_set_name
```

where *data_set_name* is the name of the new data set.

Note: If you are using a non-MVS client that does not support the SITE command, you might be able to send the SITE command to the MVS server by using the QUOTE command. For example:

```
QUOTE SITE DCBDSN=data_set_name
```

Results

Restriction: If more than one concurrent FTP user attempts to update a partitioned data set (PDS) using the FTP RENAME, DELETE, or PUT subcommands, the PDS directory might be accessed by more than one user simultaneously. This situation can cause problems with the PDS directory. To avoid this situation, when concurrent users are using FTP RENAME, DELETE, or PUT subcommands to update a PDS, use a partitioned data set extended (PDSE).

Statistics for PDS members

ISPFStats can be set to either TRUE or FALSE in a client FTP.DATA file or can be set using the LOCSITE subcommand. If ISPFStats is set to TRUE, FTP creates and maintains statistics for partitioned data set members. The following explains the effect ISPFStats has on PDS member statistics when you are issuing GET and MGET subcommands.

Note: ISPFStats is ignored for sequential data sets. Also, the record format must be either variable or fixed, and the record length must be less than 256. Transferring PDS member to PDS member in block mode or in compress mode differs in behavior from transferring in stream mode. If the user wants to preserve the statistics of a PDS member that already has the statistics and have the same statistics copied over to the target PDS member, transferring in block mode or in compress mode is required.

- **Effect of ISPFStats setting when issuing GET or MGET when the file does not already exist**

Whenever a PDS member is being transferred, FTP checks the setting of ISPFStats. If the member does not already exist, FTP follows what the ISPFStats is set to. For example:

- If ISPFStats is TRUE, FTP creates statistics for PDS members.
- If ISPFStats is FALSE, FTP does not create statistics.

- **Effect of ISPFStats setting when issuing GET or MGET when the target PDS member already exists**

Whenever a member is being transferred, FTP checks the setting of ISPFStats. If the targeted PDS member already exists, FTP considers whether the target member has statistics and the setting of ISPFStats. For example:

- If ISPFStats is TRUE and the existing member has statistics, FTP updates the statistics.
- If ISPFStats is TRUE and the existing member does not have statistics, FTP creates the statistics.
- If ISPFStats is FALSE and the existing member has statistics, FTP updates the statistics and sends a message indicating the behavior.
- If ISPFStats is FALSE and the existing member does not have statistics, FTP does not create statistics.

Generation data group support

Generation data groups (GDGs) enable you to store multiple data sets, called generation data sets (GDSs), as versions of the GDG. You cannot use FTP to create a new GDG, but you can use it to create a new version (that is, a new GDS) or to transfer an existing version of an existing GDG.

The relationship between DCBDSN and GDGs is governed by MVS allocation rules rather than FTP usage rules. Therefore, when creating a new GDG [put 'sys1.proclib(jes2)' user77.mygdg(+1)], at least one of the following must be true:

- A valid MODEL or PATTERN DSCB (for FTP, DCBDSN) specification must be coded in the FTP.DATA file when the z/OS FTP server is started.
- A valid SITE DCbdsn=*dataset_name* must be issued before a PUT command is issued.
- A data set having the same name as the GDG base must reside on the volume as the user catalog that contains the GDG definition. In this case, neither a SITE DCbdsn or a DCBDSN argument in the FTP.DATA data file is required. Allocation detects that a GDG is being created and looks in the VTOC of the volume containing the USERCATALOG for a data set (uncataloged) that has the same name as the GDG BASE (see the sample GDG JCL that follows).

Notes:

1. A model or pattern DSCB that is the same name as the GDG BASE cannot exist on an SMS managed volume. This is an SMS restriction and is documented in the DFP manuals pertaining to using data sets (generation data sets or generation data groups).
2. Allocation does not generally have any requirements about the characteristics of a MODEL DSCB (cannot be VSAM, must be on DASD). Most facilities create one model DSCB for the entire system and everyone uses that model. The system-wide model usually has no logical record length (LREcl), block size (BLKsize), record format (RECFm), data set organization (DSORG) or retention period (RETpd) associated with it.
3. The z/OS FTP server requires the MODEL DSCB to have a valid DSORG of physical sequential organization (PS). Otherwise the SITE command for the DCBDSN is ignored, and a message is issued indicating the DCBDSN was ignored.
4. GDGs are MVS-specific structures. Other operating systems might not support this structure. Using FTP to send GDG members to other operating systems is not guaranteed to yield the same results as an MVS-to-MVS transfer.
5. The RENAME subcommand does not guarantee serialization of the GDG data set. Use the PUT subcommand instead. See Informational APAR II08285 for more information.

The following restrictions apply:

- DCBDSN=USER.MYGDG(0)/ USER.MYGDG(-n), not supported
- DCBDSN=SYS1.PROCLIB(JES2), specifying a member of a PDS is not valid
- DCBDSN=SYS1.PROCLIB, valid
- The data set referenced on the DCBDSN, a DSORG of PS needed (FTP requirement)

Note: If explicit values are associated with LRECL, BLKSIZE, RECFM, or the SMS management equivalent parameters, these explicit parameters override the values associated with the model DSCB specified on the DCBDSN. The MVSGet and MVSPut subcommands configure LRECL, BLKSIZE and RECFM values which override the values that are associated with the model DSCB.

The following is a sample Job Control Language (JCL) to create a model and the GDG BASE:

```

USER77.MYGDG          -MODEL/PATTERN
VOL=SER=CPDLB1         -Volume having USERCATALOG, where USER77 is defined
(NAME(USER77.MYGDG)   -GDG BASE definition

//USER77X JOB MSGLEVEL=(1,1),MSGCLASS=D,NOTIFY=USER77
//GDGA EXEC PGM=IDCAMS
//*
//GDGMOD DD DSN=USER77.MYGDG,
//        VOL=SER=CPDLB1,
//        UNIT=SYSALLDA,
//        SPACE=(TRK,(0)),
//        DCB=(LRECL=80,RECFM=FB,BLKSIZE=6800,DSORG=PS),
//        DISP=(,KEEP)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DEFINE GENERATIONDATAGROUP -
(NAME(USER77.MYGDG) -

```



```
EMPTY -  
NOSCRATCH -  
LIMIT(255) )
```

GDG examples

Before you specify a (+nnn) value to create a new GDS, issue the following command:

```
SITE DCBDSN=model
```

This subcommand specifies an MVS data set to be used as a model. The model must have a DSORG of PS. The other DCB characteristics of the data set are not checked.

Notes:

1. Failure to have a valid DCBDSN before trying to create a new GDS might cause FTP or ALLOCATION to fail or to return unpredictable results.
2. If you issue a SITE DCbdsn LRecl BLKsize command before the creation of a new data set, the LRecl and BLKsize parameters on the SITE command override the LRecl and BLKsize parameters on the DCbdsn command.

For more information about GDGs, see [z/OS DFSMS Using Data Sets](#).

The following are sample FTP commands that access a GDG called JIMKEO.GDG.

Notes:

1. In the following examples, gdg (0), gdg (-1), and gdg (+1) specify which copy of the GDG you are using. 0 indicates the latest version, -1 indicates the previous version, and +1 indicates that a new version is created.
2. GDGALL is not supported by the z/OS FTP server. GDGALL processing occurs when the base name for the GDG is specified without a relative index value.
3. The MPUT and MGET subcommands are not applicable for GDGs.

The following example illustrates a PUT to the latest existing GDS. The working directory at the server is JIMKEO:

```
Command:  
put my.gdg gdg(0)  
>>>SITE FIXrecfm 150  
200-Blocksize must be a multiple of lrecl for FB data sets. Blocksize set to  
6150.  
200 Site command was accepted  
>>>PORT 129,34,128,245,126,229  
200 Port request OK.  
>>>STOR gdg(0)  
125 Storing data set JIMKEO.GDG.G0055V00  
250 Transfer completed successfully.  
612 bytes transferred. Transfer rate 3.24 Kbytes/sec.
```

The following example illustrates a GET of the latest GDG:

```
Command:  
get gdg(0) my.gdg2  
>>>PORT 129,34,128,245,126,233  
200 Port request OK.  
>>>RETR gdg(0)  
125 Sending data set JIMKEO.GDG.G0055V00 FIXrecfm 150  
250 Transfer completed successfully.  
612 bytes transferred. Transfer rate 3.04 Kbytes/sec.
```

The following example illustrates a PUT to a new GDS [After the STOR is complete, this new version is referenced by (0)].

```

Command:
put my.gdg gdg(+1)
>>>SITE FIXrecfm 150
200 Site command was accepted
>>>PORT 129,34,128,245,126,234
200 Port request OK.
>>>STOR gdg(+1)
125 Storing data set JIMKEO.GDG.G0056V00
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 1.16 Kbytes/sec.

```

The following example illustrates a GET of the previous GDS into the local file called my.gdg3:

```

Command:
get gdg(-1) my.gdg3
>>>PORT 129,34,128,245,126,239
200 Port request OK.
>>>RETR gdg(-1)
125 Sending data set JIMKEO.GDG.G0055V00 FIXrecfm 150
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 2.77 Kbytes/sec.

```

The following example illustrates a GET that replaces the contents of my.gdg3 with the most recent GDS:

```

Command:
get gdg(0) my.gdg3 (replace)
>>>PORT 129,34,128,245,126,243
200 Port request OK.
>>>RETR gdg(0)
125 Sending data set JIMKEO.GDG.G0056V00 FIXrecfm 150
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 3.36 Kbytes/sec.

```

The following example illustrates changing the working directory:

```

Command:
cd gdg
>>>CWD gdg
257 "'JIMKEO.GDG.'" is working directory name prefix.

```

The following example shows the files created:

```

Command:
dir
>>>MODE s
200 Data transfer mode is Stream.
>>>PORT 129,34,128,245,127,12
200 Port request OK.
>>>LIST
125 List started OK.
Volume Unit   Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
STRG73 3380K 04/30/92 1 5 FB 150 32700 PS G0003V00
STRG65 3380K 04/30/92 1 5 FB 150 32700 PS G0006V00
STRG61 3380K 04/30/92 1 5 FB 150 32700 PS G0010V00
STRG47 3380K 04/30/92 1 5 FB 150 32700 PS G0015V00
STRG47 3380K 04/30/92 1 5 FB 150 32700 PS G0021V00
STRG66 3380K 04/30/92 1 5 FB 150 32700 PS G0028V00
STRG47 3380K 04/30/92 1 5 FB 150 32700 PS G0036V00
STRG01 3380K 04/30/92 1 5 FB 150 32700 PS G0045V00
STRG53 3380K 04/30/92 1 5 FB 150 32700 PS G0055V00
STRG59 3380K 04/30/92 1 5 FB 150 32700 PS G0056V00
250 List completed successfully.
>>>MODE b
200 Data transfer mode is Block.
Command:

```

Submitting FTP requests in batch

FTP is usually run interactively by starting and entering commands from your terminal. You can also run FTP as a batch job, but you must supply the JCL file. You can use batch when you know what functions

you want to perform, when you want a hardcopy of the results, or when you want to perform an FTP function many times.

Rules: When coding the data set, file, or input stream for the ddname INPUT statement as described in [Figure 4 on page 90](#), the following rules apply:

- When you specify a data set for input as shown in [Figure 4 on page 90](#), the SEQNUMSUPPORT statement that is coded in the client FTP.DATA file determines the disposition of sequence numbers that are in the data set:
 - If the FTP.DATA file has SEQNUMSUPPORT FALSE coded (this is the default), the file, data set, or input stream that is designated in the DDNAME INPUT statement that contains the FTP commands cannot contain sequence line numbers. You must save the FTP command file as an unnumbered file.
 - If the FTP.DATA file has SEQNUMSUPPORT TRUE coded, the file, data set, or input stream that is designated in the DDNAME INPUT statement that contains the FTP commands can contain sequence numbers. These sequence numbers are removed.
- You can add comments to the command input file. The support includes standalone comment records and comments that are appended to the end of the line.

To add standalone comments, use a semicolon (;) as the first non-space character on a line. For example:

```
; This is a stand-alone comment record
```

To add a command-line comment, append a space and a semicolon to the end of the command line, followed by the comment. For example:

```
USER userx ; This is an appended comment
```

When a user ID, password, or password phrase is expected (including passwords that are required for read or write access to files or disks) and the value is intended to be retrieved from the NETRC file, the entire line must be blank. The following example shows a blank password between the user ID and a comment line.

```
USER userx  
; The line above is a blank password
```

The following example shows a blank user ID between a comment line and password.

```
; This below is a blank userid  
mypasswd
```

- If a command is too long to insert on a line, enter a plus sign (+) in place of the next command option and then enter the remaining options on the next line. For example:

```
put local_file +  
remote_file
```

- Use a blank followed by a plus sign (+) at the end of an FTP subcommand line as a continuation indicator for all FTP subcommands, except as noted under Restrictions. When the continuation indicator is encountered at the end of an FTP subcommand line, the next line is appended to the subcommand. For example, the following command is interpreted as PUT SOURCE.DS.NAME DEST.DS.NAME:

```
PUT SOURCE.DS.NAME +  
DEST.DS.NAME
```

Tips:

- You can run FTP in batch mode either by specifying data sets for input and output as in [Figure 4 on page 90](#), or without referring to data sets for input and output as in [Figure 6 on page 91](#). See [“Allocating FTP input and output data sets” on page 27](#) for the attributes that are allowed on the INPUT and OUTPUT DD statements.

- Use the EXIT parameter or code the CLIENTEXIT TRUE statement in the FTP.DATA file to display an error return code, and exit when certain errors are detected. See [FTP return codes](#) for more information.

Requirements:

- To have the FTP client perform DB/2 queries in a batch job, the DSNLOAD library must be in the link list or appear on a STEPLIB DD statement for the job.
- When connecting to a server such as a UNIX server where user IDs, passwords, directory names, and file names are case sensitive, the data in the FTP batch job must be in the correct case.

Restrictions:

- When the FTP Client API is used by an application program invoked from a batch job, the ddnames in the batch job are not available to the created FTP client process.
- You cannot use a plus sign to continue a QUOTE subcommand.
- You cannot use a plus sign to continue a password or password phrase on another line. You must take this into consideration when assigning password phrases used to log in to the FTP server from batch jobs.

See [“Logging in to FTP” on page 24](#), and each subcommand that you code in your batch job for additional information.

Figure 4 on page 90 shows an example of the JCL required to submit a batch job by referring to data sets for input and output.

```
//USER28F JOB ,CARTER,MSGLEVEL=(1,1)
//FTPSTP1 EXEC PGM=FTP,REGION=2048K,
//          PARM='9.67.112.25 (EXIT TIMEOUT 20'
//NETRC DD DSN=ANYHLQ.NETRC,DISP=SHR
//OUTPUT DD SYSOUT=H
//INPUT DD *
type e
mode b
put idss.parts
/*
```

Figure 4. JCL to run FTP in batch using data sets

Notes:

1. REGION=2048K is a minimum requirement. The requirement could increase depending on the block size of the data set being transmitted.
2. The first JCL statement is a standard job statement. The next JCL statement is an EXEC statement. It has PGM=FTP (a region parameter) because FTP might use more storage than your default region size, and a PARM field.
3. For PARM=, you can specify any parameter that is valid when invoking FTP from your terminal. See [“Using FTP” on page 19](#) for more information. These parameters are supported only on the PARM= field of the EXEC card.

As shown in [Figure 4 on page 90](#), to run FTP in batch mode, you must include the following three DD statements:

SYSPRINT DD

Alternative name for the OUTPUT DD statement. You can use SYSPRINT DD in place of the OUTPUT DD statement.

OUTPUT DD

Specifies the data set where FTP is to place the client messages and server replies generated during the FTP session.

INPUT DD

Specifies the data set where the FTP subcommands to be performed are located.

You can use the *user_id*.NETRC data set, as defined by the NETRC DD statement in [Figure 4 on page 90](#), to provide user ID, password or password phrase, and account information for a batch-processed remote

login. You can also specify the user ID, password or password phrase, and account information in the INPUT DD data set.

Figure 5 on page 91 shows the records in an INPUT DD data set that contains the FTP subcommands to be executed.

```
HOSTNAME
USERID PASSWD
DIR
PUT MYFILE.LISTING
QUIT
```

Figure 5. Contents of an INPUT DD data set

The first line of Figure 5 on page 91 contains the name of the remote host that you want FTP to log into. The second line contains the user ID followed by its password. The next three lines contain the FTP subcommands that you want FTP to perform. In this example,

- FTP lists its current working directory at the FTP server host.
- FTP sends the file MYFILE.LISTING to the server.
- FTP ends the session and disconnects from the server.

Any client messages and server replies to the subcommands you execute appear in the OUTPUT DD data set. The SYSPRINT can contain some additional messages that relate to the execution of your FTP session.

Tip: If you do not want your password or password phrase to be copied to the output file, specify your user ID, and password or password phrase on separate input lines. See Figure 6 on page 91 for an example.

Submitting requests without input and output data sets

Figure 6 on page 91 shows an easier way to submit a batch job, because you can avoid referring to data sets for input and output.

```
//USERIDX JOB USERID,MSGLEVEL=(1,1),NOTIFY=USERID,MSGCLASS=H,TIME=9
//FTP EXEC PGM=FTP,REGION=4096K
//INPUT DD *
nodeid
userid
password
CD
DIR
GET hostfile.name locfile.name
QUIT
```

Figure 6. JCL to run FTP in batch without using data sets

Figure 7 on page 91 shows step 1 creating a new GDS in batch and FTP getting the data set.

```
//USERIDX JOB USERID,MSGLEVEL=(1,1),NOTIFY=USERID,MSGCLASS=H,TIME=9
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSUT1 DD DSN=USER31.SOURCE.DATA,DISP=SHR (MYGDG.G0008V00)
//SYSUT2 DD DSN=MYGDG(+1),DISP=(,CATLG),
UNIT=SYSDA, SPACE=(TRK,(1,1)),DCB=(MODEL)
//FTP EXEC PGM=FTP,REGION=4096K
//INPUT DD *
nodeid
userid
password
CD /u/joe
DIR
GET MYGDG(0) A.DATA.SET
QUIT
//SYSPRINT DD SYSOUT=*
```

Figure 7. Job to create a new GDS in batch

Note: All the GDG allocation in batch must be complete before the start of FTP.

Submitting a batch job with concatenated files

```
//USERIDX JOB USERID,MSGLEVEL=(1,1),NOTIFY=USERID,MSGCLASS=H,TIME=9
//FTP EXEC PGM=FTP,REGION=4096K
//SYSFTPD DD DSN=SYS1.TCPPARMS(FTPCDATA),DISP=SHR
// DD * see NOTE 1
SEQNUMSUPPORT TRUE
DEBUG BAS
//INPUT DD DSN=FTP.SUBCMDS(LOGIN),DISP=SHR
// DD DSN=FTP.SUBCMDS(FTPINFO),DISP=SHR
// DD DSN=FTP.SUBCMDS(FTPCMDS1),DISP=SHR
// DD DSN=FTP.SUBCMDS(FTPCMDS2),DISP=SHR
```

Figure 8. Submitting an FTP batch job with concatenated input

Tips:

- Additional FTP parameters can be concatenated to the FTP.DATA file. This technique is useful when you need additional debugging information and you do not want to change the source FTP data file or FTP command input.
- Multiple data sets can be concatenated as input as long as you follow MVS JCL concatenation guidelines. See [z/OS MVS JCL Reference](#) for additional information. If concatenated files contain both sequenced and unsequenced input command files, each file must have a semicolon (;) in the first data column of the first record. See FTP.DATA statement SEQNUMSUPPORT information in [z/OS Communications Server: IP Configuration Reference](#) for additional information.

FTP and data set cataloging

When FTP creates a new data set, it issues a dynamic allocation request to allocate space for the data set and catalogs the data set.

If the data transfer fails when CONDDISP=DELETE, the data set is deleted and uncataloged.

If you are running a job scheduling program that detects files as they are catalogued and then schedules a subsequent job for processing, the job scheduler must take into account that setting CONDDISP=DELETE causes FTP to delete and uncatalog the data set when the file transfer fails. For generation data groups, the following might occur:

- FTP intends to create a new GDG(+1) and generates GDG.G00023V00.
- FTP of this data set fails and the GDG.G00023V00 data set is deleted and uncataloged.
- A follow-on reference for the current GDG, for example, GDG(0), would cause the data set GDG.G00022V00 to be accessed and old data to be processed.

Using the EXEC interface

The FTP EXEC interface enables you to execute FTP commands from an EXEC rather than interactively from a terminal. The FTP subcommands to be performed can be in a file (MVS data set or z/OS UNIX file), or you can code them directly in the EXEC.

By default, the FTP session dialog is printed on the terminal. If you want the dialog sent to a data set rather than the terminal do one of the following:

TSO

Specify an OUTPUT data set as part of the ALLOC statement.

z/OS UNIX System Services

Redirect the output to a z/OS UNIX file when invoking the FTP command.

The following examples are written in REXX. See the [z/OS TSO/E REXX Reference](#) and [z/OS Using REXX](#) and [z/OS UNIX System Services](#) for more information about the REXX language.

Issuing FTP subcommands from a file

Figure 9 on page 93 is an example of an EXEC that issues FTP subcommands from a data set. In this example, the FTPIN1 data set is used for the FTP subcommands, and FTPOUT1 is used to store the FTP session dialog. This example must be invoked from TSO.

```
/*REXX*/
"ALLOC DA(FTPIN1) DD(INPUT) SHR REU" /* Input will be from FTPIN1 */
if rc ~= 0 then do
  say 'Error in ALLOC INPUT, rc = ' rc
  exit
end
"ALLOC DA(FTPOUT1) DD(OUTPUT) SHR REU" /* Output goes to FTPOUT1 */
if rc ~= 0 then do
  say 'Error in ALLOC OUTPUT, rc = ' rc
  exit
end
"FTP YKTVSH" /* FTP to the YKTVSH host */
"FREE DD(INPUT)"
"FREE DD(OUTPUT)"
EXIT
```

Figure 9. How to issue the FTP subcommands from a data set

Figure 10 on page 93 is an example of an EXEC that issues FTP subcommands from a z/OS UNIX file. In this example, the name of the input file and output file are passed as arguments on the EXEC. For example,

```
EXAMPLE1 /u/user117/ftpin1 /u/user117/ftpout1
```

where EXAMPLE1 is the name of the EXEC. This example must be invoked from the z/OS UNIX shell.

```
/* rexx */
/* */
/* Input: infile - z/OS UNIX file containing FTP commands */
/* outfile - z/OS UNIX file to contain FTP output. If not specified */
/* output goes to terminal. */

parse arg infile outfile . /* get command line input */
if infile = '' then /* input file not specified */
do
  say 'Input file name is required.'
  exit 12 /* return to UNIX System Services */
end
else
  input_file = '<' infile /* redirect input from file */

if outfile <> '' then /* redirect output to file */
  output_file = '>' outfile
else
  output_file = ''

address syscall "stat (infile) fstat." /* test if input file exist */
if fstat.0 = 0 then /* input file not found */
do
  say 'Input file:' infile 'not found.'
  exit 28 /* return to UNIX System Services */
end

"ftp -v -p TCP/IP" input_file output_file /* invoke FTP client with
input and output redirection */

say "FTP client return code is:" rc /* print client return code */
exit 0 /* return to UNIX System Services */
```

Figure 10. How to issue the FTP subcommands from a z/OS UNIX file system

The following is an example of the input file (either the input data set FTPIN1 or the z/OS UNIX file /u/user117/ftpin1).

```
krasik mvsftp
cd examples
```

```

put t.info t1.info
get t1.info t2.info (r
quit

```

Where:

krasik

Is the user ID

mvsftp

Is the password

t.info

Is the file to be transferred

Issuing FTP subcommands directly from the EXEC interface

Figure 11 on page 94 is an example of how to issue FTP subcommands directly from a REXX EXEC. The REXX stack is used to hold the FTP subcommands. This example runs in both the TSO and z/OS UNIX environments.

Note: To use FTP in a z/OS UNIX environment, TSO users must be authorized users or have a default z/OS UNIX user ID.

```

/* rexx */
/* push commands on stack */
QUEUE "YKTVSH" /* server address */
QUEUE "krasik mvsftp" /* userid/password */
QUEUE "cd /tmp/examples/"
QUEUE "put t.info t1.info"
QUEUE "cd .."
QUEUE "cd dummy"
QUEUE "quit"

cmdargs = "-v -p TCP/IP" /* set ftp client arguments */

parse source . . . . . env . /* check if running under */
/* UNIX System Services */
/* env='OpenMVS' if invoked from */
/* UNIX System Services, */
/* otherwise env='' */

if env = "OpenMVS" then /* running under */
/* UNIX System Services */
/* invoke ftp client */
call bpxwunix 'ftp -v -p tcpip',stack
else /* running under TSO */
"FTP" cmdargs /* invoke ftp client */

say "FTP client return code is:" rc /* print client return code */

exit 0 /* return */

```

Figure 11. How to issue FTP subcommands from an EXEC

Note: If data set DUMMY does not exist, FTP exits with a return code.

Figure 12 on page 95 is an example of how to call FTP from a TSO/E REXX EXEC and enable it to solicit its FTP subcommands interactively from the user's TSO terminal. This requires that TSO/E has prompting enabled.


```

/* rexx */
newstack;                                /* set up a null stack for FTP*/
                                        /* so as to not disturb any */
                                        /* commands currently stacked*/
save_rexx_prompt_state = prompt();      /* save REXX prompt state */
temp = prompt('ON');                    /* set prompting on to enable */
                                        /* interactive retrieval of */
                                        /* ftp commands running */
                                        /* under TSO */
"FTP ( exit"                             /* invoke ftp client */
save_rc = rc;                           /* save FTP return code */
temp= prompt(save_rexx_prompt_state); /* restore prompt setting */
delstack;                               /* delete null stack for FTP */
return save_rc;                         /* return */

```

Figure 12. How FTP subcommands can be solicited interactively from an EXEC

Tips:

- You can call this EXEC from another TSO/E REXX EXEC by invoking this EXEC as a function call. For example, if this EXEC is named rexxftp, invoke it with the following call: `return_code = rexxftp()`
- If commands are stacked for processing and the newstack and delstack statements are removed from the sample, FTP subcommands are retrieved and processed from the stack. If no QUIT command is processed, FTP obtains commands interactively until a QUIT command is received.
- See the [FTP Client API REXX function information](#) in the *z/OS Communications Server: IP Programmer's Guide and Reference* for an alternate method of using REXX to invoke the FTP client.

FTP return codes

By default, the FTP client ignores any errors that occur during a session and exits with a return code of zero. You can direct FTP to exit on error with a nonzero return code by using one of the following methods:

- Specify the EXIT or EXIT=*nn* parameter on the FTP command
- Code the CLIENTEXIT TRUE statement in the FTP.DATA file

You can use several methods to direct the FTP client to compute return codes.

When FTP is started from the FTP Callable Application Programming Interface, all elements that compose the values described below are returned to the application. These include the client error code, the server reply code, and the FTP subcommand code. The CLIENTERRCODES setting in the FTP.DATA file has no effect on the FTP Callable API. See the *z/OS Communications Server: IP Programmer's Guide and Reference* for detailed information about using the [FTP Callable API](#).

The FTP client issues message EZA1735I to display the standard return code and the client error code when FTP is configured to exit on error, and an error occurs. The standard return code is described in [“FTP standard return codes” on page 96](#); the client error code is described in [“FTP client error codes” on page 100](#). All possible computed return codes (excluding EXIT=*nn*) can be derived from the information found in the message. EZA1735I is issued regardless of the type of return code or whether client error logging is in use. See [“FTP client error logging” on page 102](#) for more information about logging client errors.

When a critical error occurs before the client can establish its environment, FTP client initialization can exit with a return code set to client error code regardless of the type of return code requested or the EXIT=*nn* value.

Each subcommand has an EXIT_IF_ERROR flag. If you configure FTP to exit on error, the EXIT_IF_ERROR flag determines whether the FTP client exits when an error occurs. You can configure FTP to exit on error by using one of the following methods:

- Specify the EXIT or EXIT=*nn* parameter on the FTP command
- Code the CLIENTEXIT TRUE statement in the FTP.DATA file

See [“FTP subcommand codes” on page 97](#) for a list of the FTP subcommand codes and their EXIT_IF_ERROR settings.

You can use the following methods to compute return codes.

EXIT=nn

This method instructs the client to exit with a specified fixed return code for any eligible error. The EXIT parameter is specified with an equal sign (=) followed by a number in the range of 0-4095.

EXIT with CLIENTERRCODES FALSE (or unspecified) in FTP.DATA

This is the standard return code processing for FTP if you configure FTP to exit on error. This type of return code is described in [“FTP standard return codes” on page 96](#). Some limitations of this method are:

- The size of the return code might exceed the capacity (65536) of the SMF record type 30 subtype 4.
- The return code issued as a batch job step completion code generally does not match the original return code.
- The batch job step completion codes are difficult to interpret.

Because of these limitations, using one of the other available return code options is recommended. You can use this method with client error logging even if you do not configure FTP to exit on error.

See [“FTP client error logging” on page 102](#) for more information about logging client errors.

EXIT with CLIENTERRCODES TRUE in FTP.DATA

This method uses a list of error codes, defined in [“FTP client error codes” on page 100](#), to describe different types of errors that occur within the FTP client. For example, errors returned by the server are reported in the client as FTP_SERVER_ERROR. The client error codes are the same in all environments and are easier to interpret than standard return codes, but client error codes contain less information about the cause of the error.

You can use this method with client error logging even if you do not configure FTP to exit on error. See [“FTP client error logging” on page 102](#) for more information about logging client errors.

EXIT with CLIENTERRCODES EXTENDED in FTP.DATA

The EXTENDED client error code is composed of an FTP client error code and an FTP subcommand code, as described in [“FTP client error codes extended” on page 102](#). These return codes match in all environments, are easily interpreted, and provide more information regarding the cause of the error than the client error codes alone.

You can use this method with client error logging even if no EXIT parameter is in use. See [“FTP client error logging” on page 102](#) for more information about logging client errors.

FTP standard return codes

Standard FTP return codes are computed when you specify CLIENTERRCODES FALSE or the default value in the FTP.DATA file and when one of the following cases occurs:

- EXIT is specified with no parameter.
- The CLIENTEXIT FALSE statement is specified or the default value is specified in the FTP.DATA file.

Standard FTP return codes might also be displayed in the text of message EZZ9830I if LOGCLIENTERR TRUE and CLIENTERRCODES FALSE are specified in the FTP.DATA file, regardless of the following conditions:

- Whether you specified the EXIT parameter on the FTP command
- Whether you coded the CLIENTEXIT statement in the FTP.DATA file

A standard FTP return code appears in the following format:

yyxxx

Where:

yy

Is the subcommand code, a number in the range 1-99 representing the subcommand that the FTP client was executing when it detected the error. Each subcommand has an EXIT_IF_ERROR flag. If you configure FTP to exit on error, the EXIT_IF_ERROR flag determines whether the FTP client exits when an error occurs. You can configure FTP to exit on error by using one of the following methods:

- Specify the EXIT or EXIT=*nn* parameter on the FTP command
- Code the CLIENTEXIT TRUE statement in the FTP.DATA file

Table 18 on page 97 describes the possible FTP subcommand codes.

xxx

Is the reply code from the most recent reply sent by the FTP server. All FTP server replies begin with a 3-digit number. All reply codes used by the z/OS FTP server are listed the z/OS Communications Server: IP and SNA Codes. See RFC 959 and RFC 1123 for a generic description of FTP reply codes. Information about accessing RFCs can be found in [Appendix D, “Related protocol specifications,”](#) on page 381.

For example, the FTP standard return code 16550 indicates the following:

16

The GET command failed.

550

The reply code from the FTP server. The latest reply from the server began with the number 550.

The FTP standard return code 04532 indicates the following:

04

The APPEND command failed.

532

The reply code from the FTP server.

00 and 000 are valid values for *yy* and *xxx*. This means that the error occurred at a time when no FTP subcommand was being processed (*yy*=00) or at a time when no reply had been received from the server for the current process. Message EZA1735I is issued in one of the following situations:

- when the EXIT parameter is specified when the FTP client was started
- whether the CLIENTEXIT TRUE statement is specified in the FTP.DATA file

This message contains FTP standard return codes and FTP client error codes.

FTP subcommand codes

Table 18 on page 97 lists the valid FTP subcommand codes. Information in the EXIT_IF_ERROR column specifies whether an error causes FTP to end if you specify the EXIT parameter on the FTP command or specify the CLIENTEXIT TRUE statement in the FTP.DATA file.

Note: LOCSITE will EXIT_IF_ERROR only when there are no parameters on the command.

Table 18. FTP subcommand codes

| Code number | Subcommand | EXIT_IF_ERROR |
|-------------|------------------------|--|
| 00 | No subcommand selected | Determined by internal FTP CLIENT ERROR CODE |
| 1 | AMBIGUOUS | false |
| 2 | ? | false |
| 3 | ACCT | true |
| 4 | APPEND | true |
| 5 | ASCII | true |
| 6 | BINARY | true |
| 7 | CD | true |
| 8 | CLOSE | true |

Table 18. FTP subcommand codes (continued)

| Code number | Subcommand | EXIT_IF_ERROR |
|-------------|------------|---|
| 9 | TSO | false |
| 10 | OPEN | true |
| 11 | DEBUG | false |
| 12 | DELIMIT | false |
| 13 | DELETE | true |
| 14 | DIR | true |
| 15 | EBCDIC | true |
| 16 | GET | true |
| 17 | HELP | false |
| 18 | LOCSTAT | true |
| 19 | USER | true |
| 20 | LS | true |
| 21 | MDELETE | true |
| 22 | MGET | true |
| 23 | MODE | true |
| 24 | MPUT | true |
| 25 | NOOP | true |
| 26 | PASS | true |
| 27 | PUT | true |
| 28 | PWD | true |
| 29 | QUIT | true |
| 30 | QUOTE | true |
| 31 | RENAME | true |
| 32 | SENDPORT | true |
| 33 | SENDSITE | false |
| 34 | SITE | false |
| 35 | STATUS | true |
| 36 | STRUCTURE | true |
| 37 | SUNIQUE | true |
| 38 | SYSTEM | true |
| 40 | TYPE | true |
| 41 | LCD | true |
| 42 | LOCSITE | true (see previous note in “FTP subcommand codes” on page 97) |
| 43 | LPWD | false |

Table 18. FTP subcommand codes (continued)

| Code number | Subcommand | EXIT_IF_ERROR |
|-------------|------------|---------------|
| 44 | MKDIR | true |
| 45 | LMKDIR | true |
| 46 | EUCKANJI | true |
| 47 | IBMKANJI | true |
| 48 | JIS78KJ | true |
| 49 | JIS83KJ | true |
| 50 | SJISKANJI | true |
| 51 | CDUP | true |
| 52 | RMDIR | true |
| 53 | HANGEUL | true |
| 54 | KSC5601 | true |
| 55 | TCHINESE | true |
| 56 | RESTART | false |
| 57 | BIG5 | true |
| 58 | BLOCK | true |
| 59 | COMPRESS | true |
| 60 | FILE | true |
| 61 | PROXY | true |
| 62 | RECORD | true |
| 63 | SCHINESE | true |
| 64 | STREAM | true |
| 65 | GLOB | false |
| 66 | PROMPT | false |
| 67 | UCS2 | true |
| 68 | ! | true |
| 70 | DUMP | false |
| 71 | VERBOSE | false |
| 72 | CLEAR | true |
| 73 | CPROTECT | true |
| 74 | PRIVATE | true |
| 75 | PROTECT | true |
| 76 | SAFE | false |
| 77 | CCC | true |
| 78 | LANGUAGE | true |

Table 18. FTP subcommand codes (continued)

| Code number | Subcommand | EXIT_IF_ERROR |
|-------------|------------|---------------|
| 79 | FEATURE | true |
| 80 | SRESTART | true |
| 81 | AUTH | true |
| 82 | mkfifo | true |
| 83 | MVSGET | true |
| 84 | MVSPUT | true |

FTP reply codes

The z/OS FTP client displays each command it sends to the FTP server, and the FTP server reply to that command. As mandated in RFC 959, an FTP server must reply to every command an FTP client sends to it. The replies are architected to begin with a three digit reply code, followed by human readable text. The reply code indicates the status of the command; the text provides additional information. FTP clients generally interpret only the reply code and ignore the text. The reply code indicates to the client whether the server is still processing the command, or whether the server finished processing the command successfully, or whether the server stopped processing the command because of errors. For a generic description of FTP server reply codes, see RFC 959 and RFC 1123. Information about accessing RFCs can be found in [Appendix D, “Related protocol specifications,”](#) on page 381. For a complete list of the replies used by the z/OS FTP server, see [z/OS Communications Server: IP and SNA Codes](#).

FTP client error codes

[Table 19 on page 100](#) lists error codes that are used in the following situations:

- If the EXIT parameter is specified, or the CLIENTEXIT TRUE statement is specified in the FTP.DATA file, these error codes are used as condition codes for batch jobs.
- Whenever the FTP client detects one of the described errors and the FTPDATA statement CLIENTERRCODES is TRUE, these error codes are used as return codes from REXX execs.
- When CLIENTERRCODES is EXTENDED, these codes are also used to build the return code.

Message EZA1735I was issued when the EXIT parameter was specified when the client was started, or when the CLIENTEXIT TRUE statement was coded in the FTP.DATA file. This message contains FTP standard return codes and FTP client error codes.

Result: The return code 24 might supersede condition codes that were used in earlier releases.

Table 19. Client error codes

| Code | Error | Examples of cause |
|------|--------------------------|---|
| 01 | FTP_INTERNAL_ERROR | Failure to acquire storage, unexpected error in REXX stack. |
| 02 | FTP_SERVER_ERROR | Error reply returned by the server. |
| 03 | unused | N/A |
| 04 | FTP_INVALID_PARAM | Parameter specified on FTP command is not valid. |
| 05 | FTP_OPEN_IOSTREAM_FAILED | Failed to open the INPUT stream. |
| 06 | FTP_ALREADY_CONNECTED | Attempt to OPEN when already connected. |

Table 19. Client error codes (continued)

| Code | Error | Examples of cause |
|------|-------------------------|--|
| 07 | FTP_USAGE | Syntax error in a subcommand, combination of settings is not valid. |
| 08 | FTP_CONNECT_FAILED | Attempt to reach unknown host, lost connection, data connect failed. |
| 09 | FTP_TIMEOUT | Timeout waiting for response on the control or data connection. |
| 10 | FTP_SESSION_ERROR | Socket error, other send/receive errors. |
| 11 | FTP_LOGIN_FAILED | User ID, password, or account info is not valid. |
| 12 | FTP_INPUT_ERR | Error reading INPUT or STDIN. |
| 13 | FTP_INPUT_EOF | Internal use only. |
| 14 | FTP_NOTFOUND | TCP/IP stack not found, resolver not found, translation table not found or could not be loaded. |
| 15 | FTP_INVALID_ENVIRONMENT | Missing INPUT DD. |
| 16 | FTP_NOT_ENABLED | Improper installation of TCP/IP. |
| 17 | FTP_AUTHENTICATION | Security authentication or negotiation failure, incorrect specification of security keywords. |
| 18 | FTP_FILE_ACCESS | Data set allocation failure, recall failure, open failure. |
| 19 | FTP_FILE_READ | File corrupted. |
| 20 | FTP_FILE_WRITE | Out of space condition, close failure. |
| 21 | FTP_CONVERSION | Error during data translation or setup not otherwise specified. |
| 22 | FTP_PROXY_ERR | Error during proxy processing not otherwise specified. |
| 23 | FTP_SQL_ERR | Error returned by the SQL process, including connect failure. |
| 24 | FTP_CLIENT_ERR | Other errors in the client, some unrecoverable interface errors. |
| 25 | FTP_EOD_BEFORE_EOF | In a block mode transfer, the last record did not include the EOF marker. In a stream mode transfer, the last record received before the connection was closed did not end with a <CRLF> (carriage return followed by line feed) sequence. |
| 26 | FTP_NEEDS_CONNECTION | Possible causes include using a subcommand that requires a connection to the server when no connection exists. |

Table 19. Client error codes (continued)

| Code | Error | Examples of cause |
|------|----------------------------|---|
| 27 | FTP_EXIT_EZAFCCMD_PREVENT | User exit EZAFCCMD rejects the command. |
| 28 | FTP_EXIT_EZAFCCMD_TERM | User exit EZAFCCMD ends the client. |
| 29 | FTP_EXIT_EZAFCCMD_WRONG_RC | The FTP client ends because of an invalid return code from user exit EZAFCCMD. |
| 30 | FTP_EXIT_EZAFCREP_TERM | User exit EZAFCREP ends the client. |
| 31 | FTP_EXIT_EZAFCREP_WRONG_RC | The FTP client ends because of an invalid return code from user exit EZAFCREP . |

Guideline: Codes 27 - 31 are caused by a user exit that is installed on this system. Contact your local system programmer for assistance in determining why the action was rejected.

FTP client error codes extended

The EXTENDED client error codes feature is enabled by specifying CLIENTERRCODES EXTENDED in the FTP.DATA data set used by the client. EXTENDED implies TRUE, meaning that client error codes are used for return code generation, and it supplements the information returned. See the following format of the return code:

ecyy

Where:

ec

The client error code set by the FTP client. See [“FTP client error codes” on page 100](#) for more information.

yy

The subcommand code, which is a number in the range of 0 - 99. See [“FTP subcommand codes” on page 97](#) for a list of the subcommand codes.

The return code returned to the user or batch job is the 4-digit ecyy value. This value can also be derived from the contents of message EZA1735I, which displays the standard return code and client error code.

FTP client error logging

If you activate FTP client error logging, message EZZ9830I provides the subcommand code, last reply code, and computed return code of any error that causes FTP to exit. Configure FTP to exit on error by using one of the following methods:

- Specify the EXIT or EXIT=nn parameter on the FTP command
- Code the CLIENTEXIT TRUE statement in the FTP.DATA file

Otherwise, client error logging drives message EZZ9830I.

The client error logging feature is enabled by specifying LOGCLIENTERR TRUE in the FTP.DATA data set used by the client.

Each subcommand has an EXIT_IF_ERROR flag. If you configure FTP to exit on error, the EXIT_IF_ERROR flag determines whether the FTP client exits when an error occurs. You can configure FTP to exit on error by using one of the following methods:

- Specify the EXIT or EXIT=nn parameter on the FTP command
- Code the CLIENTEXIT TRUE statement in the FTP.DATA file

See [“FTP subcommand codes” on page 97](#) for a list of FTP subcommand codes and their EXIT_IF_ERROR settings.

If the client session is interactive, the message is displayed on the user terminal. If the client session is not running in an interactive environment, the message appears in the system log and in the batch job log. The information contained in EZZ9830I includes:

- The address space name.
- The FTP subcommand code.
- The last reply code from the server (or 000 if none).
- Information about whether EXIT was specified when the client was started or the CLIENTEXIT TRUE statement was specified in the FTP.DATA file.
- The type of computed return code.
- The computed return code value based on the start parameters and configuration settings. It might not match the actual return code observed from the client.

Because EZZ9830I is written to the system log when the FTP client is running as a batch job, the message can be used to drive automation. The computed return code displayed in the message might be the EXIT=*nn* value, a standard return code, a client error code, or a client error code extended.

The standard return code can be derived from the FTP subcommand code and last reply from the server displayed in the message text. See *z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM)* for the complete message format of EZZ9830I and an explanation of the computed return code value and the other fields in the message.

Restarting a failed data transfer

About this task

FTP supports two subcommands for restarting data transfers that fail. Select the appropriate subcommand based on the data transfer mode at the time the data transfer failed.

- If stream mode data transfer was used, use the `srestart` subcommand to restart the data transfer. See [“SRestart subcommand—Restart a stream data transfer” on page 276](#) for a description of stream mode restart.
- If block or compressed mode was used, use the `restart` subcommand to restart the data transfer. See [“REStart subcommand - Restart a checkpointed data transfer” on page 242](#) for a description of block or compressed mode restart.

To restart a block or compressed mode data transfer, checkpointing must have been initiated prior to the start of the transfer. You can use either the `LOCSITE` or `SITE` subcommands to initiate checkpointing.

- The `LOCSITE` subcommand parameter `CHKPTINT` controls the checkpoint interval at the client, and the parameters `CHKPTPREFIX` and `RESTGET` control the naming of the client checkpoint data set.
- The `SITE` subcommand parameters `CHKPTINT` and `RESTPUT` control the checkpoint and restart processing at the server.

Each of the `LOCSITE`/`SITE` parameters is also supported in the `FTP.DATA` file.

If the local file or the remote file is a z/OS UNIX named pipe, do not use the `srestart` or `restart` subcommands to restart the file transfer. If the transfer type is binary and the local file is the named pipe, you can resume the transfer using the restart instructions below, or you can use the instructions below to start the transfer again. If the transfer type is not binary, or if the remote file is the named pipe, you must start the transfer again.

Procedure

- To start the transfer again:
 - If you are trying to restart a file transfer into a named pipe, use these steps to restart it:
 1. Stop the named pipe reader, directing it to discard the data it has received so far.

2. Empty the named pipe. This prevents FTP from appending to any residual data in the named pipe.
 3. Restart the named pipe reader.
 4. Repeat the original file transfer subcommand.
- If you are trying to restart a file transfer from a named pipe, use these steps to restart it:
 1. Stop the named pipe writer.
 2. Empty the named pipe. This prevents the named pipe writer from appending to any residual data in the named pipe.
 3. Restart the named pipe writer, having it re-create the data stream.
 4. Issue the original file transfer subcommand.
 - To restart the file transfer (for binary transfers only):
 - If you were trying to restart a transfer into a named pipe in the FTP client file system, take these steps to resume the file transfer:
 1. Figure out the number of bytes, *bytes*, the named pipe reader on the FTP client host read from the named pipe.
 2. Ensure that the named pipe is empty.
 3. From the z/OS FTP client, issue QUOTE REST *bytes* to direct the server to start the next file transfer from offset *bytes* into the server file.
 4. Issue the original file transfer subcommand.
- Note:** The z/OS FTP server supports the REST command in stream mode only when EXTENSIONS REST_STREAM is coded in the server FTP.DATA.
- If you were trying to restart a transfer from a named pipe in the FTP client file system, follow these steps to resume the file transfer:
 1. Do one of the following from the FTP client to learn the number of bytes, *bytes*, stored in the server file:
 - Issue the DIr subcommand
 - Issue the LS -l subcommand
 - Issue QUOTE SIZE *serverfile*
 2. Ensure that the named pipe is empty.
 3. Restart your named pipe writer, telling it to regenerate the data starting from point bytes.
 4. Issue the APpend subcommand to append the named pipe to the server file.
- Note:** The z/OS FTP server supports the SIZE command only for regular z/OS UNIX files, and only when the EXTENSIONS SIZE statement is coded in the server FTP.DATA.

Using z/OS UNIX System Services named pipes

The z/OS FTP client and server support named pipes (or FIFO files) in the z/OS UNIX System Services file system. The terms named pipe and FIFO file are synonymous. A named pipe can be specified as an argument on the following subcommands:

- APpend
- DELEte
- DIr
- Get
- LOCSItE, (when the CHMOD parameter is specified)
- LS
- MDelete

- MGet
- MKFifo
- MPut
- Put
- RENAME
- SItE, (when the CHMOD parameter is specified)

The following FTP configuration options for client and server are related to sending data from and receiving data into the named pipe:

- FIFOIOTIME
- FIFOOPENTIME
- UNIXFILETYPE

Before you start a transfer to or from a named pipe, you must configure the following on the named pipe host:

- FILETYPE=SEQ (the default value)
- UNIXFILETYPE=FIFO

FTP uses default values for FIFOIOTIME and FIFOOPENTIME if you do not configure these values.

You can configure these options by coding statements in the FTP.DATA file. For the z/OS FTP client, you can also use the LOCSItE subcommand to configure these options. Alternatively, for the z/OS FTP server, you can configure these values by doing the one of the following:

- Issuing the SItE subcommand from the z/OS FTP client
- Sending SITE commands to the server from any FTP client

When you transfer data from a named pipe to another host using FTP, the following apply:

- FTP cannot open the named pipe until another process on the named pipe host opens the named pipe for reading. If FTP is the first process to open the named pipe, it is blocked until the read process opens the named pipe, or until the FIFOOPENTIME timer expires.
- The data is permanently removed from the named pipe, rather than being copied from the named pipe.

When you use FTP to transfer data to a named pipe, the following conditions apply:

- FTP cannot open the named pipe until another process on the named pipe host opens the named pipe for reading. If FTP is the first process to open the named pipe, it is blocked until the read process opens the named pipe, or until the FIFOOPENTIME timer expires.
- FTP always appends to any existing data in the named pipe. You cannot use FTP to replace the contents of a named pipe.
- FTP applies the UNICODEFILESYSTEMBOM setting to inbound transfers when it is storing Unicode data into the named pipe. In cases when FTP would store a byte order mark (BOM) into a regular file, FTP appends a BOM byte sequence to existing data. If the BOM byte sequence appears outside of the first character position, it is interpreted as a zero-width nonbreaking space character. You must take this into consideration when you code UNICODEFILESYSTEMBOM.

Guidelines:

- If FTP is the only process writing to the named pipe, and you are sending all data to the named pipe with a single transfer, you can configure any value for UNICODEFILESYSTEMBOM.
- If your application can tolerate a superfluous zero-width nonbreaking space character in the named pipe, you can configure any value for UNICODEFILESYSTEMBOM.
- You can always configure UNICODEFILESYSTEMBOM=NEVER.
- If your application requires a BOM in the named pipe, but cannot tolerate a zero-width nonbreaking space character, consider setting UNICODEFILESYSTEMBOM=NEVER after the first transfer into the named pipe.

For information about the UNICODFILESYSTEMBOM configuration option, see [UNICODFILESYSTEMBOM \(FTP client and server\)](#) UNICODFILESYSTEMBOM (FTP client and server) information in [z/OS Communications Server: IP Configuration Reference](#).

The z/OS operating system does not provide serialization for named pipes. Multiple processes can read from or write to the same named pipe concurrently. When a process reads from a named pipe, data is removed from the named pipe. The same data is not presented to the other processes that are reading from the pipe. When a process writes to a named pipe, it is possible that the data it writes will be interleaved with data written by other processes. You must take this into consideration when planning to use z/OS UNIX named pipes.

Named pipes in the z/OS UNIX file system for the client

When files exist as named pipes in the z/OS Unix file system, you can change their file permissions using the LOCSite chmod subcommand.

You can use the following z/OS FTP subcommands to transfer data into named pipes in the client file system:

- Get
- MGet

The named pipe can exist before you issue Get or MGet subcommands or the FTP client can create the named pipe for you as part of Get and MGet subcommand processing.

You can use the following z/OS FTP subcommands to send data from existing named pipes in the client z/OS UNIX file system:

- APpend
- Put
- MPut

You must configure UNIXFILETYPE=FIFO to transfer data to or from a named pipe in the FTP client file system.

When UNIXFILETYPE=FIFO is configured on the client, all z/OS UNIX files that are created by the client during Get and MGet processing are created as named pipes. The FTP client always requests file permissions 777 when it creates a named pipe, but the configured UMASK value determines the actual file permissions. See the [UMASK \(FTP client and server\)](#) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UMASK statement and [“LOCSite subcommand—Specify site information to the local host” on page 179](#) for information about configuring the UMASK value.

Unlike most types of z/OS UNIX files, data written to a named pipe is always appended to existing data rather than replacing existing data. You cannot specify the REPLACE parameter on Get and MGET subcommands when UNIXFILETYPE=FIFO is configured.

Unlike most types of z/OS UNIX files, data read from a named pipe is removed from the named pipe instead of being copied from the named pipe. The client must read from a named pipe to send it to the server. Sending a named pipe empties the named pipe.

Restrictions:

- The z/OS operating system does not provide serialization for named pipes. Multiple processes on the client can read from or write to the same named pipe at one time.
- Restarting of file transfers to or from named pipes is not allowed.
- You cannot specify the REPLACE option when you transfer data into a named pipe.

Results: The following results apply when the z/OS FTP client stores a file as a named pipe in the z/OS UNIX file system:

- The client waits up to the number of seconds specified by the FIFOOPEN TIME configuration option to open the named pipe for writing. The FTP client is unable to open the named pipe until the process

that reads from the named pipe opens the pipe. This is true even when FTP created the named pipe. A large FIFOOPEN TIME value gives you more time to start the named pipe reader, but could cause the data connection to time out if the client waits the entire length of time specified by the FIFOOPEN TIME value.

If you are issuing the MGet subcommand, the client blocks for the length of time specified by the FIFOOPEN TIME configuration option for each local named pipe that is not opened for reading by a process on the FTP client host at the time of transfer.

- The client waits up to the number of seconds specified by the FIFOIOTIME configuration option for each write to the named pipe to complete. In general, the client does not block during writes unless it writes to the named pipe much faster than the named pipe reader reads from the pipe. If the client cannot write any data to the named pipe for the number of seconds specified by the FIFOIOTIME configuration option, it fails the file transfer.

The following are the results of sending data from a named pipe in the z/OS UNIX file system:

- The client waits up to the number of seconds specified by the FIFOIOTIME configuration option to open the named pipe for reading. The FTP client is not able to open the named pipe until the process that writes to the named pipe opens the pipe. A large FIFOOPEN TIME value gives you more time to start the named pipe writer, but could cause the data connection to time out if the client waits the entire length of time specified by the FIFOOPEN TIME value.

If you are issuing the MPut subcommand, the client blocks for the number of seconds specified by the FIFOIOTIME configuration option for each local named pipe that is not opened for writing by a process on the FTP client host at the time of transfer.

- The client waits up to the number of seconds specified by the FIFOIOTIME configuration option for each read from the named pipe to complete. The client can block if the pipe write process stops writing to the named pipe but does not close it. If the FTP client cannot read any data from the named pipe during the number of seconds specified by the FIFOIOTIME value, it fails the file transfer.

Steps to save a file as a named pipe in the z/OS FTP client UNIX file system

Before you begin

You must start an application that can read from the named pipe, and it must open the named pipe, before FTP can transfer data into the named pipe.

Procedure

1. Do one of the following to set the UNIXFILETYPE FIFO value at the client host:
 - Code a UNIXFILETYPE statement in the client FTP.DATA file before starting the FTP daemon. See the UNIXFILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UNIXFILETYPE statement.
 - Issue a LOCStite subcommand with the UNIXFILETYPE parameter after starting the FTP client. See [“LOCStite subcommand—Specify site information to the local host” on page 179](#) for more information about the LOCStite subcommand.
2. Do one of the following to set the FILETYPE=SEQ value at the client host:
 - Code the FILETYPE statement in the client FTP.DATA file before you start the FTP client. See the FILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the FILETYPE statement.
 - Issue the LOCStite subcommand with the FILEtype parameter after starting the client. See [“LOCStite subcommand—Specify site information to the local host” on page 179](#) for more information about the LOCStite subcommand.

Tip: FILETYPE=SEQ is the default value for FILETYPE. You can issue the `locstat filetype` subcommand from the z/OS FTP client to determine whether you need to reset the FILETYPE value:
3. Optional: Do one of the following to set the FIFOOPEN TIME and FIFOIOTIME values at the client:

- Code the FIFOOPEN TIME and FIFOIOTIME statements in the client FTP.DATA file before you start the FTP daemon. See [z/OS Communications Server: IP Configuration Reference](#) for more information about the FIFOOPEN TIME and FIFOIOTIME statements.
- Issue a LOC SItE subcommand with the FIFOOPEN TIME and FIFOIOTIME configuration options after you start the client. See [“LOC SItE subcommand—Specify site information to the local host” on page 179](#) for more information about the LOC SItE subcommand.

The FTP client uses default values for FIFOOPEN TIME and FIFOIOTIME parameters if you do not configure these values explicitly.

4. On the FTP client host, start the process that reads from the named pipe.
5. Use the Get subcommand to transfer a file from the client to the server, specifying a target file in the z/OS UNIX file system. For example, `get local_named_pipe remote_file`.

Results

You know that you have completed these steps correctly when the following events occur:

- The client receives reply 226 or 250 from the server, which indicates that the server sent the file.
- The client issues message EZA2536I or EZA1617I to indicate the total number of bytes that it received.

Tips:

- You do not need to create the named pipe with FTP before you initiate the transfer. FTP creates the named pipe during file transfer if the named pipe is not already in the file system. However, FTP is not able to transfer data into a named pipe that it creates until another process on the FTP client host opens the named pipe for reading.
- If the FTP client creates the named pipe for you, the configured umask value determines the file permissions. See the UMASK parameter information in [“LOC SItE subcommand—Specify site information to the local host” on page 179](#), or the UMASK (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UMASK statement.

Steps for sending data from a named pipe in the z/OS FTP client UNIX file system

Before you begin

- You must start an application that can write from the named pipe, and it must open the named pipe, before FTP can transfer data into the named pipe.
- You must create the named pipe. See [MKFIFO command information in z/OS UNIX System Services Command Reference](#) for more details.

Procedure

1. Do one of the following to set the UNIXFILETYPE FIFO value at the client host:
 - Code a UNIXFILETYPE statement in the client FTP.DATA file before you start the FTP daemon. See the UNIXFILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UNIXFILETYPE statement.
 - Issue a LOC SItE subcommand with the unixfiletype parameter after you start the FTP client. See [“LOC SItE subcommand—Specify site information to the local host” on page 179](#) for more information about the LOC SItE subcommand.
2. Do one of the following to set the FILETYPE=SEQ value at the client host:
 - Code the FILETYPE statement in the client FTP.DATA file before you start the FTP client. See the FILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the FILETYPE statement.

- Issue the LOCSite subcommand with the FILEtype parameter after you start the client. See [“LOCSite subcommand—Specify site information to the local host” on page 179](#) for more information about the LOCSite subcommand.

Tip: SEQ is the default value for the FILETYPE configuration option. You can use the `locstat filetype` subcommand from the z/OS FTP client to determine whether you need to reset the FILETYPE value.

3. Optional: Do one of the following to set the FIFOOPENIME and FIFOIOTIME values at the client:

- Code the FIFOOPENIME and FIFOIOTIME statements in the client FTP.DATA file before starting the FTP daemon. See [z/OS Communications Server: IP Configuration Reference](#) for more information about the FIFOOPENIME and FIFOIOTIME statements.
- Issue the LOCSite subcommand with the FIFOOPENIME and FIFOIOTIME parameters after you start the client. See [“LOCSite subcommand—Specify site information to the local host” on page 179](#) for more information about the LOCSite subcommand.

The FTP client uses default values for the FIFOOPENIME and FIFOIOTIME configuration options if you do not configure these values explicitly.

4. On the FTP client host, start the process that writes to the named pipe.

5. Issue the Put subcommand to transfer a file from the client to the server. Specify the named pipe as the local file as follows:

```
put local_named_pipe remote_file
```

Results

You know you have completed these steps correctly when both of the following events occur:

- The client receives reply 226 or 250 from the server, which indicates that the server received the file.
- The client issues message EZA2536I or EZA1617I, which indicate the total number of bytes that it sent.

Named pipes in the server z/OS UNIX file system

When files are stored in the server z/OS UNIX file system as named pipes, you can delete, rename, and list the named pipes using these z/OS FTP subcommands:

- DELEte
- DIr
- LS
- MDelete
- REName

With other FTP clients, you can specify a named pipe as the argument of these commands: DELE, RNFR, RNTD, LIST, NLST.

Using the z/OS FTP client, you can create a named pipe in the server z/OS UNIX file system with this subcommand:

- MKFifo

From other FTP clients, you can use the QUOTE subcommand to send an XFIF command to the remote host:

- QUOTE XFIF <pathname>

When you configure UNIXFILETYPE FIFO at the FTP server, all files you send to the server z/OS UNIX file system are stored as named pipes, and you can retrieve data from named pipes in the server z/OS UNIX file system.

You can use these z/OS FTP subcommands to retrieve from named pipes in the FTP server file system:

- Get
- MGet

From other FTP clients, use the RETR command to retrieve data from named pipes.

The FTP server can retrieve data only from existing named pipes. You can create the named pipe as described earlier in this section, or another process on the FTP server host can create the named pipe.

You can use these z/OS FTP subcommands to send files to named pipes in the server z/OS UNIX file system:

- APpend
- Put
- MPut

Use the SUnique subcommand to set store-unique off before using the Put or MPut subcommand to send files to named pipes.

From other FTP clients, you can use APPE or STOR commands to store a file as a named pipe in the server file system.

Unlike most types of z/OS UNIX files, data written to a named pipe is always appended to existing data rather than replacing existing data. Therefore, the STOR command is equivalent to the APPE command when UNIXFILETYPE=FIFO is configured. You cannot replace a named pipe by issuing the STOR command as you can with other types of files in the z/OS UNIX file system.

Unlike most types of z/OS UNIX files, data read from a named pipe is removed from the named pipe permanently. Retrieving data from a named pipe in the server file system destroys the contents of the named pipe.

Restrictions:

- Anonymous users are not allowed to read from or write to named pipes in the z/OS FTP server z/OS UNIX file system.
- The STOU (Store-unique) command is not allowed when UNIXFILETYPE=FIFO is configured.
- The z/OS operating system does not provide serialization for named pipes. Multiple processes on the server can read from or write to the same named pipe concurrently.
- Restart of file transfers to or from named pipes is not allowed.

Results: The following results apply when the server stores a file as a named pipe in the z/OS UNIX file system:

- The server waits up to the number of seconds specified by the FIFOOPEN TIME configuration option to open the named pipe for writing. The FTP server cannot open the named pipe until the process that reads from the named pipe opens the pipe. A large FIFOOPEN TIME value gives you more time to start the named pipe reader, but could cause the data connection to time out if the server waits for the number of seconds specified by the FIFOOPEN TIME value.

If you issue the MPut subcommand, the server blocks for the number of seconds specified by the FIFOOPEN TIME value for each remote named pipe that is not opened for reading by a process on the FTP server host at the time of transfer.

- The server waits up to the number of seconds specified by the FIFOIOTIME value for each write to the named pipe to complete. In general, the server does not block during writes unless it writes to the named pipe much faster than the named pipe reader reads from the pipe. If the server does not write any data to the named pipe for the number of seconds specified by the FIFOIOTIME value, it fails the file transfer.

The following results apply when the server retrieves data from a named pipe in the z/OS UNIX file system:

- The server waits up to the number of seconds specified by the FIFOOPEN TIME value to open the named pipe for reading. The FTP server cannot open the named pipe until the process that writes to the named pipe opens the pipe. A large FIFOOPEN TIME value gives you more time to start the named pipe writer, but could cause the data connection to time out if the server waits for the number of seconds specified by the FIFOOPEN TIME value.

If you issue the MGet subcommand, the server blocks for the number of seconds specified by the FIFOOPEN TIME value for each remote named pipe that is not opened for writing by a process on the FTP server host at the time of transfer.

- The server waits up to the number of seconds specified by the FIFOIOTIME value for each read from the named pipe to complete. The server might block if the pipe write process stops writing to the named pipe but does not close it. If the FTP server cannot read any data from the named pipe for the number of seconds specified by the FIFOIOTIME value, it fails the file transfer.

Steps for storing a file as a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client

Before you begin

You must start an application that can read from the named pipe, and it must open the named pipe, before FTP can transfer data into the named pipe.

Procedure

1. Optional: Create the named pipe on the server host:

- a) Issue the SItE subcommand to configure the server UMASK value.

For example: `site UMASK=<mask>`

You can also configure the server UMASK value by coding the UMASK statement in the FTP.DATA file. See the UMASK (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UMASK statement.

Tip: After you create the named pipe, you can change the file permissions by issuing the SItE subcommand with the CHMOD parameter.

- b) Issue the the MKFifo subcommand to create the named pipe at the server.

For example: `mkfifo <pathname>`

See [“MKFifo subcommand—Create a named pipe at the FTP server host”](#) on page 219 for information about the MKFifo subcommand.

Tips:

- You do not need to create the named pipe with FTP before initiating the transfer. FTP creates the named pipe during file transfer if the named pipe is not already in the file system, or another process on the server host can create the named pipe for you.
- After you create a named pipe, you can display and manipulate it with the following FTP subcommands:
 - DELEte
 - DIr
 - Ls
 - REName
 - SItE subcommand with CHMOD parameter

2. Do one of the following to set the UNIXFILETYPE FIFO value at the server host:

- Code a UNIXFILETYPE statement in the server FTP.DATA file before you start the FTP daemon. See the UNIXFILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UNIXFILETYPE statement.
- Issue the SItE subcommand with the UNIXFILETYPE parameter after you log in to the server. See [“SItE subcommand—Send site-specific information to a host ”](#) on page 247 for more information about the SItE subcommand.

3. Do one of the following to set the FILETYPE=SEQ value at the server host:

- Code the FILETYPE statement in the server FTP.DATA file before you start the FTP daemon. See the FILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the FILETYPE statement.
- Issue the SItE subcommand with the FILEtype parameter after you log in to the server. See FILETYPE statement information in [“SItE subcommand—Send site-specific information to a host”](#) on page 247 for more details about the SItE subcommand.

Tip: FILETYPE=SEQ is the default FILETYPE value. You can use the stat (*filetype* subcommand from the z/OS FTP client to determine whether you need to reset the FILETYPE value.

4. Optional: Do one of the following to set the FIFOOPEN TIME and FIFOIOTIME values at the server:

- Code FIFOOPEN TIME and FIFOIOTIME statements in the server FTP.DATA file before you start the FTP daemon. See [z/OS Communications Server: IP Configuration Reference](#) for more information about the FIFOOPEN TIME and FIFOIOTIME statements.
- Issue the SItE subcommand with the FIFOOPEN TIME and FIFOIOTIME parameters after you log in to the server. See [“SItE subcommand—Send site-specific information to a host”](#) on page 247 for more information about the SItE subcommand.

The FTP server uses default values for the FIFOOPEN TIME and FIFOIOTIME configuration options if you do not configure these values explicitly.

5. On the FTP server host, start the process that reads from the named pipe.

6. Issue the APpend subcommand to send the file from the client to the server, specifying a target file in the z/OS UNIX file system as follows: append *localFile named_pipe*

Results

You know that you have completed these steps correctly when the following occurs:

- The server sends reply 226 or 250 to the client to indicate that it received the file successfully.
- The client issues message EZA2536I or EZA1617I to indicate the total number of bytes that were sent.

Guidelines: Use these guidelines for using any FTP client to store a file as a named pipe in the z/OS FTP server UNIX file system:

- You can issue the QUOTE subcommand to send a SITE command with the UMASK, FILETYPE UNIXFILETYPE, FIFOOPEN TIME, and FIFOIOTIME parameters to the FTP server, or to send an XFIF command to the server. See [“Steps for storing a file as a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client”](#) on page 111 for details.
- You can use the APPE or STOR commands to send data to a named pipe on the server.

Tip: When you send data to a named pipe on the z/OS server, the STOR command is treated the same as an APPE command.

Steps for retrieving data from a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client

Before you begin

You must start an application that can write to the named pipe, and it must open the named pipe, before FTP can transfer data from the named pipe.

Procedure

1. Create the named pipe on the server host.

You can use FTP to create the named pipe, or another process on the server host can create the named pipe.

To create the named pipe using FTP:

- a. Issue the SItE subcommand to configure the server UMASK value. For example, site UMASK=<mask>.

You can also configure the server UMASK value by coding the UMASK statement in FTP.DATA. See the [UMASK \(FTP client and server\) statement information in z/OS Communications Server: IP Configuration Reference](#) for more details about the UMASK statement.

Tip: After you create the named pipe, you can change the file permissions by issuing the SItE subcommand with the CHMOD parameter.

- b. From the z/OS FTP client, issue the MKFifo subcommand to create the named pipe. For example, mkfifo <pathname>.

See [“MKFifo subcommand—Create a named pipe at the FTP server host” on page 219](#) for information about the MKFifo subcommand.

Tip: After you create a named pipe, you can display and manipulate it with the following FTP subcommands:

- DELEte
- DIr
- Ls
- REName
- SItE subcommand with the CHMOD parameter

2. Do one of the following to set the UNIXFILETYPE FIFO value at the server host:

- Code a UNIXFILETYPE statement in the server FTP.DATA file before you start the FTP daemon. See the [UNIXFILETYPE \(FTP client and server\) statement information in z/OS Communications Server: IP Configuration Reference](#) for more details about the UNIXFILETYPE statement.
- Issue the SItE subcommand with the UNIXFILETYPE parameter after you log in to the server. See [“SItE subcommand—Send site-specific information to a host ” on page 247](#) for more information about the SItE subcommand.

3. Do one of the following to set the FILETYPE configuration option to the value SEQ at the server host:

- Code the FILETYPE statement in the server FTP.DATA file before you start the FTP daemon. See the [FILETYPE \(FTP client and server\) statement information in z/OS Communications Server: IP Configuration Reference](#) for more details about the FILETYPE statement.
- Issue the SItE subcommand with the FILEtype parameter after you log in to the server. See [“SItE subcommand—Send site-specific information to a host ” on page 247](#) for more information about the SItE subcommand.

Tip: SEQ is the default value for the FILETYPE configuration option. You can issue the stat (*filetype* subcommand from the z/OS FTP client to determine whether you need to reset the FILETYPE value.

4. Optional: Do one of the following to set FIFOOPENTIME and FIFOIOTIME values at the server:

- Code FIFOOPENTIME and FIFOIOTIME statements in the server FTP.DATA file before you start the FTP daemon.

See [z/OS Communications Server: IP Configuration Reference](#) for more information about the FIFOOPENTIME and FIFOIOTIME statements.

- Issue the SItE subcommand with the FIFOOPENTIME and FIFOIOTIME parameters after you log in to the server.

See [“SItE subcommand—Send site-specific information to a host ” on page 247](#) for more information about the SItE subcommand.

The FTP server uses the default values for the FIFOOPENTIME and FIFOIOTIME configuration options if you do not configure these values explicitly.

5. On the FTP server host, start the process that writes to the named pipe.
6. Issue the Get subcommand to retrieve data from the named pipe.

For example, get *<named pipe>*.

Results

You know that you have completed these steps correctly when both of the following events occur:

- The server sends reply 226 or 250 to the client to indicate that it sent the file successfully.
- The client issues message EZA2536I or EZA1617I to indicate the total number of bytes that it received.

Guidelines: Use these guidelines for using any FTP client to retrieve from a named pipe in the z/OS FTP server UNIX file system:

- You can issue the QUOTE subcommand to send a SITE command with the UMASK, UNIXFILETYPE, FIFOOPTIME, and FIFOTIME parameters to the FTP server, or to send an XFIF command to the server. See [“Steps for storing a file as a named pipe in the z/OS FTP server UNIX file system using the z/OS FTP client” on page 111](#) for details.
- Issue the RETR command to retrieve data from a named pipe on the server.

Interfacing with JES

The MVS Job Entry System (JES) enables you to perform the following functions:

- Submit jobs (consisting of JCL and data) to the job scheduler for execution
- Spool JCL messages and SYSOUT during execution
- Print the output
- View the output
- Delete job output

FTP server provides the following functions in its JES interface:

- Submitting a job
- Displaying the status of all the user's jobs
- Receiving the spool output of the job (JCL messages and SYSOUT)
- Deleting a job
- Submitting a job and automatically receiving output
- Terminating access to JES

Steps for submitting a job

A job consists of job control language (JCL) and data. You can use FTP to submit a job.

Procedure

Perform the following steps to submit a job using FTP.

1. Create the JCL and data that you want to submit, using the editor on your client.
 - If the FTP server is set up for JESINTERFACELEVEL 1, to be able to display the status, receive spool output for, and delete a job, the job name in the JCL must be the USERIDx, where x is a 1-character letter or number and USERID must be the user ID you use to log in to the FTP server to submit the job. Otherwise if the job name in the JCL is not USERIDx, the job can be submitted but the DIR subcommand does not display the job, and the GET and DELETE subcommands will not be supported for the job.
 - If the FTP server is set up for JESINTERFACELEVEL 2, the job name can be any name you are authorized to view using the Security Authorization Facility (SAF), such as RACF. See [“JESINTERFACELEVEL differences” on page 121](#) for more information on security and JESINTERFACELEVEL 2.

You can determine how the FTP server is set up and whether you have authority to view jobs by entering the STAT client command and the SITE command. For JESINTERFACLEVEL 2, the STAT command returns:

```
211-JESINTERFACLEVEL is 2
211-JESOWNER is USER1
211-JESJOBNAME is USER1*
```

If the SITE JESOWNER= completes successfully, then the user has SAF authority to other users' jobs. If the SITE JESJOBNAME= completes successfully, then the user has SAF authority to other jobs.

Note: The maximum LRecl for the submitted job is 254 characters. JES scans only the first 72 characters of JCL.

-
2. Start a session with the FTP server on the MVS system to which you want to submit the job.

-
3. After you have logged into the server, specify that you want to interface to JES with a site parameter by entering the following:

```
SITE FILEtype=JES
```

-
4. To submit the JCL file you have created, enter the following:

```
PUT filename filetype
```

Results

When you are done, the JCL is then submitted to the JES internal reader and waits for an initiator to start the job. The job is submitted under the user ID that you used when you logged on to the system unless a different user ID is specified on the JOB card.

The default for *filetype* is SEQ, and when you want to go back to normal FTP file transfer mode, enter the following:

```
SITE FILEtype=SEQ
```

See [z/OS MVS JCL Reference](#) for more information about using JCL.

Result: If the submitted JCL does not specify the MSGCLASS value on the JOB statement, the default value sysout class A is used.

Displaying the status of a job

This section describes client operation when the MVS server has been placed in FILEtype=JES mode with the SITE command. After you have submitted your job, you can determine whether it is waiting for execution, running, or finished. The status of all the jobs that are on the JES spool for your user ID can be displayed. The format of the display depends on the value of JESINTERFACLEVEL.

DIR output with JESINTERFACLEVEL=1

The following is a sample of the DIR output with JESINTERFACLEVEL=1:

```

MYUSRIDA JOB05444 OUTPUT 3 spool Files
MYUSRIDB JOB05766 OUTPUT 6 spool Files
MYUSRIDC JOB05832 OUTPUT 6 spool Files
MYUSRIDD JOB05946 ACTIVE
MYUSRIDE JOB06021 INPUT

```

1 2 3 4

1. The first column displays the job name.
2. The second column displays the job ID, assigned by JES. This 8-character job ID, consisting of the word JOB followed by a 5-digit number or the letter J followed by a 7-digit number, is assigned by JES to identify your job.
3. The third column displays the status of the job.

The status is one of the following:

INPUT

The job was received, but not run yet. If the JCL specified that the job be put on hold, it appears with this status.

ACTIVE

The job is running.

OUTPUT

The job has finished and has output to be printed or retrieved. For each OUTPUT job, there are spool files that consist of JCL messages, JES messages, initiator and terminator messages, and SYSOUT. For jobs with a status of OUTPUT, the number of spool files for each job is specified in the DIR display output.

4. The fourth part of the display lists the number of retrievable spool files for the job.

Note: The LS subcommand lists the job ID without the job status or number of spool files. Providing this spool information consumes a lot of computer resources. Use the LS subcommand rather than the DIR subcommand when the job status and number of spool files are not required.

DIR command with JESINTERFACELEVEL=2

The following is a sample of the DIR command with JESINTERFACELEVEL=2:

```

dir
EZA1701I >>> PORT 127,0,0,1,4,5
200 Port request OK.
EZA1701I >>> LIST
125 List started OK for JESJOBNAME=USER1*, JESSTATUS=ALL and JESOWNER=USER
EZA2284I JOBNAME JOBID OWNER STATUS CLASS
EZA2284I USER1A JOB00083 USER1 OUTPUT A ABEND=806 3 spool files
EZA2284I USER1 JOB00070 USER1 OUTPUT A RC=000 5 spool files
EZA2284I USER1J JOB00082 USER1 OUTPUT A (JCL error) 3 spool files
EZA2284I USER1 JOB00054 USER1 OUTPUT A RC=0000 5 spool files
EZA2284I USER1D JOB00029 USER1 OUTPUT A RC=0000 5 spool files
EZA2284I USER1C JOB00028 USER1 OUTPUT A RC=0000 5 spool files
EZA2284I USER1B JOB00027 USER1 OUTPUT A RC=0000 5 spool files
EZA2284I USER1 TSU00024 USER1 ACTIVE TSU
EZA2284I USER1W JOB00081 USER1 INPUT A -DUP-
EZA2284I USER1p JOB00093 USER1 INPUT A -HELD-
250-JESEENTRYLIMIT of 10 reached. Additional entries not displayed
250 List completed successfully.
EZA1460I Command:
dir tsu00024
EZA1701I >>> PORT 127,0,0,1,4,6
200 Port request OK.
EZA1701I >>> LIST tsu00024
125 List started OK for JESJOBNAME=USER1*, JESSTATUS=ALL and JESOWNER=USER
EZA2284I JOBNAME JOBID OWNER STATUS CLASS
EZA2284I USER1 TSU00024 USER1 ACTIVE TSU
-----
EZA2284I STEPNAME ++++++++ PROCNAME OS390R5
EZA2284I CPUTIME 0.503 ELAPSED TIME 3853.718
250 List completed successfully.
EZA1460I Command:
dir job54
EZA1701I >>> PORT 127,0,0,1,4,7
200 Port request OK.
EZA1701I >>> LIST job54
125 List started OK for JESJOBNAME=USER1*, JESSTATUS=ALL and JESOWNER=USER
EZA2284I JOBNAME JOBID OWNER STATUS CLASS
EZA2284I USER1 JOB00054 USER1 OUTPUT A RC=000
EZA2284I -----
EZA2284I ID STEPNAME PROCSTEP C DDNAME BYTE-COUNT
EZA2284I 001 JESE H JESMSG LG 1200
EZA2284I 002 JESE H JESJCL 526
EZA2284I 003 JESE H JESYSMSG 1255
EZA2284I 004 STEP57 H SYSUT2 741
EZA2284I 005 STEP57 A SYSPRINT 209
EZA2284I 5 spool files
250 List completed successfully.
EZA1460I Command:

```

- For a list of jobs these fields are displayed:

JOBNAME

The job name.

JOBID

The job ID, assigned by JES. This 8-character job ID, consisting of the word JOB followed by a 5-digit number or the letter J followed by a 7-digit number, is assigned by JES to identify your job.

OWNER

The user ID that owns the job.

STATUS

The current job status. The status is one of the following:

INPUT

The job was received, but not run yet.

HELD

The JCL specified that the job is to be put on hold.

ACTIVE

The job is running.

OUTPUT

The job has finished and has output to be printed or retrieved. For each OUTPUT job, there are spool files that consist of JCL messages, JES messages, initiator and terminator messages, and

SYSOUT. For jobs with a status of OUTPUT, the number of spool files for each job is specified in the DIR display output.

CLASS

The job class.

These fields are followed by information about the job termination (if it has been reached) or clarification of the status field. This information might include an ABEND code, an indication of a JCL error, an indication that the job is held as a DUP, or the maximum return code for the job in a 4-digit decimal field.

- For a TSO user the following additional fields are displayed:

STEPNAME

The procstepname for the logon procedure.

PROCNAME

The logon procedure.

CPUTIME

The total CPU seconds used.

ELAPSED TIME

The total elapsed time in seconds.

- For an individual job, these additional fields are displayed for each spool file:

ID

The ID for the spool file being listed.

STEPNAME

The job step that generated the spool file.

PROCSTEP

The procedure stepname (if any) that generated the spool file.

C

The SYSOUT class of the spool file.

DDNAME

The DDNAME associated with this spool file.

BYTE-COUNT

The total number of bytes in the spool file.

Receiving spool output

You can retrieve JCL messages, JES messages, initiator and terminator messages, and SYSOUT data sets either individually or as a group.

Notes:

1. In JES2 (if JESINTERFACELEVEL=1), the spool files retrieved by GET and tallied by DIR must be in a hold queue (commonly class=H).
2. In JES3 (if JESINTERFACELEVEL=1), the spool files must be in a hold queue reserved for external writers. Ask your system programmer for the class that says (HOLD=EXTWTR) in the JES3 installation stream.
3. The maximum record length that can be received at the server is 254 characters before the record is truncated.
4. Receiving the output of a job does not remove the job output from the queue. To remove the job output from the queue, you must issue a DELETE command.

Receiving individual spool files

Retrieving the spool files one at a time enables you to see whether a job ran correctly before you retrieve the rest of the output, giving you greater control over retrieving job information. If your FTP server is configured with JESINTERFACELEVEL=2, then a DIR command displays the output job completion code

and the number and size of the job files. If JESINTERFACELEVEL=1, a DIR command displays the number of the output job files.

To retrieve the spool output while FILEtype=JES is specified, specify the job ID and either the number of the spool file or the JES data set name of the spool file that you want.

You can specify a short form of the job ID by entering the letter J followed by a 4-digit or 5-digit job number. For example:

```
GET JOB05444.1 JOB05444.FILE1 (REPLACE
GET JOB05766.6 ASSEMBLY.FILE6
GET JOB06235.2 (REPLACE
GET J6235.USER1.USER1.TSU00072.D0000002.JESMSGLG OUTPUT
GET JOB00275.4
GET J7438.3
```

In these examples, *foreign_file* is specified first, followed by *local_file* (on your client workstation) with the appropriate options, such as REPLACE. The first example requests that the first spool file for JOB05444 be transmitted and replace the file on your client named JOB05444.FILE1. The second command requests that the sixth spool file for JOB05766 be transmitted to your client with the name ASSEMBLY.FILE6. The fourth command has differing results depending on the JESGETBYDSN value that is configured. You can specify the JESGETBYDSN value as an option on the SITE subcommand or in the server FTP.DATA file, or accept the default value JESGETBYDSN FALSE.

If JESGETBYDSN FALSE is configured, the file named JOB06235.USER1.USER1.TSU00072.D0000002.JESMSGLG of job JOB06235 on the server host is submitted to JES. The resulting output is saved on the client in file OUTPUT. If the JESINTERFACELEVEL 2 statement is coded in the server FTP.DATA file and either the JESGETBYDSN option is specified on the SITE subcommand or JESGETBYDSN TRUE is coded in the server FTP.DATA file, the spool file with the JES data set name USER1.USER1.TSU00072.D0000002.JESMSGLG on the server host is stored on the client host in file OUTPUT. The JES spool file data set name has the same format as an MVS data set name and is case sensitive. You can find this data set using the SDSF utility on the Job Data Set (JDS) panel. Even though the JES spool file data set name has the same format as an MVS data set name, the name can be longer than an MVS data set name and can allow different character sets; it should not be enclosed in quotation marks. See *z/OS SDSF Operation and Customization* for more information about JES data set names. See [“Steps for submitting a job and automatically receiving output” on page 120](#) for more information about the behavior of the JESGETBYDSN parameter.

If you have specified FILEtype=JES, you can use the MGET subcommand to receive output from multiple jobs without specifying them one at a time. For example, you can enter:

```
MGET parameter
```

The FTP client requires an MGET subcommand parameter. The parameter is passed to the FTP server but is not used. The server returns all of the SYSOUT files for all the jobs in the HELD queue for your user ID for FTP servers configured for JESINTERFACELEVEL=1. For JESINTERFACELEVEL=2, all jobs that match the filters JESJOBNAME, JESOWNER and JESSTATUS are retrieved whether they are in the HELD queue or not. Note that the JESJOBNAME and JESOWNER SITE parameters allow wildcards (*,?). When using the MGET subcommand the JESOWNER value should be OUTPUT.

Notes:

1. On an MVS FTP server, *local_file* must be specified.
2. Truncation can cause a loss of data.
3. A GET command performed on an empty data set erases the contents of the existing local data set.
4. Receiving the output of a job does not remove the job output from the queue. To remove the job output from the queue, you must issue a DELETE command.

Receiving a group of spool files

To retrieve all the spool files associated with the same job simultaneously into the same destination file, specify:

```
GET jobid.x
```

where *x* can be uppercase or lowercase. All the spool files are transferred together and put into file *jobid.x*. The following line appears between each retrieved JES spool file:

```
!! END OF JES SPOOL FILE !!
```

This enables you to easily find the end of each spool file.

You can also specify a data set name to send the files to, such as:

```
GET jobid.x data_set_name
```

All the spool files are put into the file named *file.name*. This eliminates the need to retrieve each spool file separately. For example, GET J3456.X retrieves all the spool files for JOB03456 and puts them in a file named J3456.X.

The command MGET with any parameter produces the same results as issuing GET *jobid.x* commands for each job that is associated with your user ID.

Deleting a job

You can delete a job before or during execution, or you can delete the output of a job before you have retrieved it.

Delete a job by using the DELETE subcommand while in the FILEtype=JES mode and the job ID. You can specify either the 8-character job ID or a short form of the job ID by entering the letter J followed by a 1 to 7-digit job number. For example,

```
DELETE JOB05444  
DELETE J3672
```

When you issue the DELETE command, all spool output related to a job is deleted.

The host returns the message CANCEL SUCCESSFUL after it deletes the job.

Steps for submitting a job and automatically receiving output

You can submit a job by using the JCL that you have built on the FTP server rather than on the FTP client. Automatic retrieval of jobs works only if the file contains a single job. It does not work for files that include more than one job (multiple JOB cards).

Procedure

Perform the following steps to submit a job using FTP.

1. Create the JCL and data that you want to submit and save it on the MVS host where the FTP server resides.

The JCL can reside in a sequential data set, or partitioned data set, or z/OS UNIX file. If JESINTERFACELEVEL=1, then the job name in the JCL must be USERID*x*, where *x* is a 1-character letter or number. Additionally, the output class for any data sets you want to retrieve (MSGCLASS or SYSOUT files) contained in your JCL must specify a JES HOLD output class. If JESINTERFACELEVEL=2, then the JESJOBNAME and JESJOBOWNER must match the jobname and jobowner. Additionally, JESSTATUS must be set to ALL or OUTPUT and the logged in FTP user ID must have access to nodeid.userid.jobname.jobid.

-
2. Start a session with the FTP server on the MVS system to which you want to submit the job.

-
3. After you have logged into the server, specify that you want to interface to JES with a site parameter by entering the following code:

```
SITE FILEtype=JES NOJESGETBYDSN
```

4. Submit the JCL file you have created by entering the following command:

```
GET SOURCE.JCLNAME OUTPUT.DSNAME
```

The *OUTPUT.DSNAME* defines the data set at the FTP client site that is to contain the HELD job output when the job completes.

Results

The MVS FTP server reads the data set *SOURCE.JCLNAME* and submits it to the JES internal reader.

You know you are done when you see the following replies. The next two replies indicate that the job has been submitted and the MVS FTP server is waiting for it to complete. The third reply indicates that the job has finished and the files have been copied to your output data set:

```
125 Submitting job SOURCE.JCLNAME FIXrecfm 80
125 When JOB05125 is done, will retrieve its output250 Transfer completed
successfully.
```

Note: When submitting a job and automatically receiving the output, remember that your session is suspended. You should use care, based on the anticipated run time of your job, when using this function. If your session times out, you must restart FTP and manually retrieve your output. Session timeouts are caused by the following cases:

- The FTP Server does not wait long enough for the job that is executing to end. Increase the JESPUTGETTO interval in the FTP.DATA data statement on the server. This defaults to 10 minutes and defines the amount of time FTP waits for the submitted job to complete before timing out.
- The FTP client does not wait long enough for the job to complete and the server to retrieve the output. Increase the DATACTTIME timer value in the client. This defaults to two minutes and defines the amount of time the client waits for a response from the server.
- The control or data connection is closed. This is usually caused by a firewall that timed out the session because of inactivity. Add FTPKEEPALIVE (control connection) and DATAKEEPALIVE (data connection) statements in the FTP.DATA data file.
- FTP client and FTP Server receive resets. This is usually caused by a firewall that timed out the session because of a lack of activity. Add an FTPKEEPALIVE statement or decrease the time interval on the current FTPKEEPALIVE statement in the FTP.DATA data file. The keepalive value on FTPKEEPALIVE must be less than the expected value of the server.

Terminating access to JES

The FTP default for FILEtype is SEQ. When you want to end access to JES and return to FTP in its normal file transfer mode, specify the following:

```
SITE FILEtype=SEQ
```

JESINTERFACELEVEL differences

The FTP JESINTERFACELevel 2 allows increased functionality for users over that available with JESINTERFACELevel 1. Jobs allowable by the system (JESSPOOL RACF class) whether they are in held or nonheld output class can be viewed. Information about jobs such as held/dup for jobs on the internal reader, CPU time of running jobs, and number of sysout data sets for completed jobs can be displayed.

If the user does a DIR subcommand without any operands, the display shows matches of the search criteria — one line per task. If the user does a DIR subcommand for a specific JOBID, the status of the job

is listed and, if the status is OUTPUT, it lists information about all the sysout data sets created by this job (and available in the JES spool data set at the time of access). If a JOBID is supplied and the job is active, it lists information about the job step name, cpu busy, and elapsed time.

Whenever a DIR subcommand is executed (with or without arguments), the JESJOBNAME, JESENTRYLIMIT, JESOWNER and JESSTATUS filter keywords can be used to limit the number of entries that are returned to the user. A DIR subcommand with all JES keywords set to their default and JESOWNER set to USER1 returns information about all batch jobs that have USER1 registered as OWNER. If the value of JESOWNER is set to empty (space), it defaults to the logged-in user ID.

- To list jobs for all users, a value of JESOWNER=* must be specified. If the value of JESJOBNAME is set to empty (space), it defaults to the logged-in user ID suffixed with an asterisk (if user is USER1, then JESJOBNAME defaults to USER1*).
- To list all job names, a value of JESJOBNAME=* must be specified. If the value of JESSTATUS is set to empty, it defaults to all types of output.
- To list only the completed jobs, specify a value of JESSTATUS=OUTPUT.

The subcommands DIR, LS, MDELETE, and MGET allow wildcard (*) filtering on a specific jobid only if JESINTERFACELevel 2 is active. A DIR subcommand results in the use of the LIST FTP command. An LS, MDELETE, or MGET subcommand results in the use of the NLST FTP command.

The following table summarizes the behavior of the different levels of JESINTERFACELevel. The term *matching filters* means all jobs that match parameters of JESJOBNAME, JESOWNER, JESSTATUS, and up to the JESENTRYLIMIT entries.

| Subcommand | JESINTERFACELevel 1 | JESINTERFACELevel 2 |
|------------------------|---|---|
| DIR [*] | All jobs | All Jobs matching filters |
| DIR Jxx | All jobs (Jxx ignored) | Details for Jxx (*) |
| LS | All jobids | All jobids matching filters |
| LS Jxx with wildcards | All jobids | All Jobs matching filters (*) |
| LS Jxx no wildcards | All jobids (Jxx ignored) | All Jxx.nnn sysout ds names matching filters (*) |
| GET Jxx.1 [local-file] | Return single sysout file | Return single sysout file |
| GET Jxx.x [local-file] | All Jxx sysout files in one file with separators | All Jxx sysout files in one file with separators |
| MGET * | One file per job - each file with separators - local file names default to jobids | One file per job matching filters - each file with separators - local files names default to jobids |
| MGET jxx | As MGET * (Jxx ignored) | One file per job matching filters - each file with separators - local file names default to Jxx.nnn (*) |
| DELETE Jxx | Jxx deleted | Jxx Deleted |
| DELETE Jxx.1 | Not supported | Not supported |
| MDELETE * | All jobs deleted | All jobs matching filters deleted |
| GET jclds outfile | Server jclds submitted, all output returned in outfile with separators | Server jclds submitted, all output returned in outfile with separators |

JESINTERFACELevel 2 returns different entries from JESINTERFACELevel 1 with wildcard (*) filtering, assuming the default values are used for the JESOWNER, JESJOBNAME, and JESENTRYLimit plus JESSTATUS=OUTPUT. The format of the returned data might also be different.

FILETYPE=JES parameters on DIR, LS and MGET subcommands allow an asterisk (*) for all job IDs, or a specific job ID. If a job ID is specified, but no such job ID exists, an error reply is returned to the client.

The NLST command is used under the covers for MGET and MDELETE to obtain lists of the resource names to issue individual GET or DELETE subcommands against. For job IDs, the list is simply a list of JES job IDs. For sysout data set IDs, the list is a list of JES job IDs suffixed with sysout data set numbers — in the same syntax that is supported by the GET and DELETE subcommands.

LIST, NLST, MDELETE, and MGET all show different behaviors for JESINTERFACELevel 2 over JESINTERFACELevel 1. This is because LIST and NLST use the filters for JESINTERFACELevel 2, and the commands DIR, LS, MDELETE, and MGET use LIST and NLST.

- For job IDs, an NLST command returns:

```
JOB00013
JOB00034
STC00067
```

- For sysout data set IDs, an NLST command returns:

```
JOB00013.1
JOB00013.2
JOB00013.3
JOB00013.4
```

JES security

If FTP.DATA does not set the JESINTERFACELevel to 2, the FTP server behaves as in releases prior to z/OS Communications Server level V2R10. FTP clients are allowed to submit jobs to JES, retrieve held output that matches their logged in user ID plus one character, and delete held jobs that match their logged in user ID plus one character.

If JESINTERFACELevel is set to 2, then FTP clients have the ability to retrieve and delete any job in the system permitted by the Security Access Facility (SAF) resource class JESSPOOL. For that reason, JESINTERFACELevel=2 should be specified only if the proper JES and SDSF security measures are in place to protect access to JES output. The SAF controls used for JESINTERFACELevel=2 are essentially a subset of those used by SDSF. Therefore, if an installation has customized SAF facilities for SDSF, then they are configured for FTP JES level 2.

Before customizing the FTP-to-JES interface, you must complete JES customization. For example, JESJOBS is an SAF class that controls which users can submit jobs to JES. JESSPOOL is the SAF class that controls which users can access output jobs. Customize these SAF classes before beginning customization of the FTP-to-JES interface.

JESSPOOL defines resource names as <nodeid>, <userid>, <jobname>, <Dsid>, <dsname>. An FTP client can delete an output job if it has ALTER access to the resource that matches its nodeid, userid, and job name. If the FTP client has READ access to the resource, it can list, retrieve, or GET the job output. (JESINTERFACELevel 2 uses the SAPI interface to JES, so READ authority is required to list job status or retrieve job output.) See the [z/OS JES2 Initialization and Tuning Guide](#) for more information on JES security. See [z/OS MVS Using the Subsystem Interface](#) for more information on the SAPI interface.

Changing JESSTATUS, JESOWNER, and JESJOBNAME

There are three filters used by the FTP server to control the display of jobs:

- JESSTATUS
- JESOWNER
- JESJOBNAME

JESSTATUS can be changed by an FTP client using the SITE command to filter jobs in INPUT, ACTIVE, or OUTPUT state. The SDSF resources checked for these states are *ISFCMD.DSP.INPUT.jesx*, *ISFCMD.DSP.ACTIVE.jesx*, and *ISFCMD.DSP.OUTPUT.jesx*, respectively. At login time (USER command), the default value is set to ALL if READ access is allowed to all three classes. Otherwise it attempts to set

it to OUTPUT, ACTIVE, and then INPUT if the appropriate READ access is allowed. If no READ access is allowed to any of the classes, JESSTATUS is set to OUTPUT but JESOWNER and JESJOBNAME cannot be changed from the default. In this way, SAF controls can be put in place to limit FTP clients to whatever status of jobs an installation requires.

At login time, JESOWNER has the value of the logged in user ID. Authority to change JESOWNER is obtained by READ access to RACF profiles *ISFCMD.FILTER.OWNER*. An FTP client with READ access to *ISFCMD.FILTER.OWNER* is allowed to change the JESOWNER parameter using the SITE command.

At login time, JESJOBNAME has the value of the logged in user ID plus an asterisk (*). Authority to change JESJOBNAME is obtained by READ access to RACF profile *ISFCMD.FILTER.PREFIX*. An FTP client with READ access to *ISFCMD.FILTER.PREFIX* is allowed to change the JESJOBNAME parameter using the SITE command.

If a user is not authorized to the appropriate *ISFCMD.DSP.<status>.jesx*, any SITE JESSTATUS command is rejected with a reply:

```
200 User xxxxxxxx is not authorized to filter on JESSTATUS, JESSTATUS remains  
xxxxxxx
```

If a user is not authorized to filter on JESOWNER, any SITE JESOWNER command is rejected with a reply:

```
200 User xxxxxxxx is not authorized to filter on JESOWNER, JESOWNER remains  
xxxxxxx
```

If a user is not authorized to filter on JESJOBNAME, any SITE JESJOBNAME command is rejected with a reply:

```
200 User xxxxxxxx is not authorized to filter on JESJOBNAME, JESJOBNAME remains  
xxxxxxx*
```

Displaying the status of jobs (LIST and NLST)

No security checks are made when returning information to the user on an NLST or LIST (DIR, MGET, or MDELETE) command that has a wild card character of * or ? for a jobid. The filter variables were reset when the user connection was made. Any attempts to change them caused the appropriate security check to be made.

When a user issues a DIR command for a specific jobid, a check for READ access to JESSPOOL resource *nodeid.userid.jobname.jobid* is made. This check is performed to prevent a different user (one who happens to know a jobid of another user's job) from obtaining status information, even though the user was not allowed to filter on OWNER. If a user is not allowed READ access, the following reply is sent:

```
550 RETR fails: nodeid.ownerid.jobname.jobid. User not authorized.
```

Browsing of SYSOUT data sets

When a SYSOUT data set is requested for transfer to the FTP client, the server checks JESSPOOL resource *nodeid.userid.jobname.jobid.Dsid.dsname* for READ access. If a user is not allowed READ access to the resource, the following reply is sent:

```
550 RETR fails: nodeid.ownerid.jobname.jobid. User not authorized.
```

Deleting/purging of SYSOUT data sets

When an FTP client requests to delete a SYSOUT data set, the server checks JESSPOOL resource *nodeid.userid.jobname.jobid.Dsid.dsname* for ALTER access. If a user is not allowed ALTER access to the resource, the following reply is sent:

```
550 DELE fails: nodeid.ownerid.jobname.jobid. User not authorized.
```

JES examples

The following example shows the JCL file USER121.JCL.CNTL(SMFALL) being submitted to the JES. Before FTP commands are issued, only the data set USER121.FTP.EXAMPLE exists on MVSXA2.

```
User: ftp 9.67.113.24 621
System:
      IBM FTP CS V1R5
      FTP: using TCPCS
      FTP.DATA FILE NOT FOUND.  USING HARDCODED DEFAULT VALUES.
      Connecting to 9.67.113.24, port 621
      220-FTPSERVE IBM FTP CS V1R2 at MVSVIC03.TCP.RALEIGH.IBM.COM,
19:03:
      on 2003-01-17
      220 Connection will close if idle for more than 5 minutes.
      NAME (<host>:tsouserid):
User: user121
System:
      >>>USER user121
      331 Send password please.
      Password:
      >>>PASS *****
      230 user121 is logged on. Working directory is "/u/user121".
      Command:
```

```
User: site file=jes
System:
      >>>SITE file=jes
      200 Site command was accepted
      Command:
User: put 'user121.jcl.cntl(mvsjob)'
System:
      >>>SITE FIXrecfm 80 LRECL=80 RECFM=FB BLKSIZE=27920
      200 Site command was accepted
      >>>PORT 9,67,112,25,4,37
      200 Port request OK.
      >>>STOR 'user121.jcl.cntl(mvsjob)'
      125 Sending Job to JES Internal Reader FIXrecfm 80
      250-It is known to JES as JOB02189.
      250 Transfer completed successfully.
      1066 bytes transferred in 3.118 seconds. Transfer rate 0.34 Kbytes/sec.
      Command:
```

```
User: dir
System:
      >>>PORT 9,67,112,25,4,38
      200 Port request OK.
      >>>LIST
      125 List started OK.
      USER121A JOB00067 INPUT
      250 List completed successfully.
      Command:
User: dir
System:
      >>>PORT 9,67,112,25,4,39
      200 Port request OK.
      >>>LIST
      125 List started OK.
      USER121A JOB00067 ACTIVE
      250 List completed successfully.
      Command:
User: dir
System:
      >>>PORT 9,67,112,25,4,40
      200 Port request OK.
      >>>LIST
      125 List started OK.
      USER121A JOB00067 OUTPUT      4 Spool Files
      250 List completed successfully.
      Command:
```

```

User: lcd 'user121.ftp.example.'
System:
      Local directory name set to USER121.FTP.EXAMPLE.
      Command:
User: lpwd
System:
      Local directory is USER121.FTP.EXAMPLE.
      Command:
User: dir
System:
      >>>PORT 9,67,112,25,4,41
      200 Port request OK.
      >>>LIST
      125 List started OK.
      USER121A JOB00067 OUTPUT 4 Spool Files
      250 List completed successfully.
      Command:
User: get job00067.x spoolall
System:
      >>>PORT 9,67,112,25,4,42
      200 Port request OK.
      >>>RETR job00067.x
      125 Sending all SP00L files for requested JOBID.
      250 Transfer completed successfully.
      5935 bytes transferred in 4.755 seconds. Transfer rate 1.25 Kbytes/sec.
      Command:

```

```

User: get job00067.1 spool1
System:
      >>>PORT 9,67,112,25,4,43
      200 Port request OK.
      >>>RETR job00067.1
      125 Sending data set USER121.USER121A.JOB00067.D000002.JESMSGGLG
      250 Transfer completed successfully.
      1962 bytes transferred in 0.739 seconds. Transfer rate 2.65 Kbytes/sec.
      Command:
User: get job00067.2 spool2
System:
      >>>PORT 9,67,112,25,4,44
      200 Port request OK.
      >>>RETR job00067.2
      125 Sending data set USER121.USER121A.JOB00067.D000003.JESYSMSG
      250 Transfer completed successfully.
      1982 bytes transferred in 2.123 seconds. Transfer rate 0.93 Kbytes/sec.
      Command:
User: get job00067.3 spool3
System:
      >>>PORT 9,67,112,25,4,45
      200 Port request OK.
      >>>RETR job00067.3
      125 Sending data set USER121.USER121A.JOB00067.D000004.JESYSMSG
      250 Transfer completed successfully.
      1982 bytes transferred in 2.123 seconds. Transfer rate 0.93 Kbytes/sec.
      Command:
User: get job00067.4 spool4
System:
      >>>PORT 9,67,112,25,4,46
      200 Port request OK.
      >>>RETR job00067.4
      125 Sending data set USER121.USER121A.JOB00067.D000103.?
      250 Transfer completed successfully.
      1227 bytes transferred in 0.380 seconds. Transfer rate 3.23 Kbytes/sec.
      Command:

```



```

User: get job00067.5 spool5
System:
    >>>PORT 9,67,112,25,47
    200 Port request OK.
    >>>RETR job00067.5
    550 Index 5 is greater than number of spool files for JOB00067
    Command:
User: dir
System:
    >>>PORT 9,67,112,25,4,50
    200 Port request OK.
    >>>LIST
    125 List started OK.
    user121A JOB00067 OUTPUT 4 Spool Files
    250 List completed successfully.
    Command:
User: delete job00067
System:
    >>>DELE job00067
    250 Cancel Successful
    Command:
User: dir
System:
    >>>PORT 9,67,112,25,4,51
    200 Port request OK.
    >>>LIST
    125 List started OK.
    No jobs found on Held queue
    250 List completed successfully.
    Command:
User: site filetype=seq
System:
    >>>SITE filetype=seq
    200 Site command was accepted
    Command:
User: quit
System:
    >>>QUIT
    221 Quit command received. Goodbye.
    READY

```

After executing the FTP commands, the following data sets now exist on MVSXA2:

```

USER121.FTP.EXAMPLE.SPOOLALL
USER121.FTP.EXAMPLE.SPOOL1
USER121.FTP.EXAMPLE.SPOOL2
USER121.FTP.EXAMPLE.SPOOL3
USER121.FTP.EXAMPLE.SPOOL4

```

Note: In most situations, the INPUT status is too fast to be captured by issuing DIR. However, if the ACTIVE or OUTPUT status of the job is captured, the INPUT status has been passed successfully.

The following are examples displayed by the shown DIR command.

DIR with JESJOBName=* JESOwner=* JESSTATUS=ALL JESENTRYLimit=200 shows all jobs in the system.

```

DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A
USER1A   JOB00001  USER1    INPUT   A -HELD-
USER1A   JOB00002  USER1    INPUT   A -DUP-
USER2B   JOB00022  USER2    ACTIVE  D STEP=STEPNAME PROC=PROCSTEP CPUT= 7.27 ELAPT= 7.27
*MASTER* STC00002  +MASTER+ ACTIVE  STEP=          PROC=          CPUT= 7.48 ELAPT= 7.48
USER3A   JOB00061  USER3    OUTPUT  D 3 spool files RC=0000
USER4A   JOB00070  USER4    OUTPUT  D 17 spool files ABEND=0806

```

DIR with JESJOBName=* JESOwner=USER1 JESSTATUS=ALL JESENTRYLimit=200 shows all jobs owned by USER1

```

DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A
USER1A   JOB00001  USER1    INPUT   A -HELD-
USER1A   JOB00002  USER1    INPUT   A -DUP-

```

DIR with JESJOBName=* JESOwner=USER14 JESSTATUS=ALL JESENTRYLimit=200 shows all jobs owned by USER14

```

DIR
250 No tasks found for JESJOBName=*, JESSTATUS=ALL and JESOwner=USER14

```

DIR T* with JESJOBName=* JESOwner=USER1 JESSTATUS=ALL JESENTRYLimit=200 shows all TSO jobs.

```

DIR T*
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A

```

DIR with JESJOBName=* JESOwner=* JESSTATUS=ALL JESENTRYLimit=5 shows the first 5 jobs in the system

```

DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A
USER1A   JOB00001  USER1    INPUT   A -HELD-
USER1A   JOB00002  USER1    INPUT   A -DUP-
USER2B   JOB00022  USER2    ACTIVE  D STEP=STEPNAME PROC=PROCSTEP CPUT= 7.27 ELAPT= 7.27
*MASTER* STC00002 +MASTER+ ACTIVE    STEP=          PROC=          CPUT= 7.48 ELAPT= 7.48
JESENTRYLIMIT of 5 reached. Additional entries not displayed

```

If JESINTERFACELevel is set to 2, the DIR or LIST command for a specific jobid allows you to display specific SYSOUT data sets.

DIR JOB000061 with JESJOBName=* JESOwner=* JESSTATUS=ALL JESENTRYLimit=5 displays SYSOUT data sets from JOB000061.

```

DIR JOB000061
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER3A   JOB00061  USER3    OUTPUT  D 3 spool files RC=0000
          ID  STEPNAME  PROCSTEP  C DDNAME  REC-COUNT  COMMENT
          001 JESE             H JESMSG LG      18
          002 JESE             H JESJCL      11
          003              A JESYSMSG      22

```

If JESINTERFACELevel is set to 2, then the GET command should produce results similar to those below (JESJOBName=* JESOwner=* JESSTATUS=ALL JESENTRYLimit=200 displays SYSOUT data sets from JOB000061).

```

DIR JOB00061
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER3A   JOB00061 USER3     OUTPUT  D 3 spool files RC=0000
          ID  STEPNAME PROCSTEP C DDNAME  REC-COUNT COMMENT
          001 JESE             H JESMSG LG      18
          002 JESE             H JESJCL  11
          003                 A JESYSMSG 22

DIR 'JOB00061.1'
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER3A   JOB00061 USER3     OUTPUT  D 3 spool files RC=0000
          ID  STEPNAME PROCSTEP C DDNAME  REC-COUNT COMMENT
          001 JESE             H JESMSG LG      18

GET JOB00061.1
125 Sending data set USER3.USER3A.JOB00061.D0000002.JESMSG LG
250 Transfer completed successfully.
1012 bytes transferred in 0.40 seconds. Transfer rate 25.30 Kbytes/sec.
GET 'JOB00061.1' (REP
125 Sending data set USER3.USER3A.JOB00061.D0000001.JESMSG LG
250 Transfer completed successfully.
1012 bytes transferred in 0.40 seconds. Transfer rate 25.30 Kbytes/sec.

```

If JESINTERFACELevel is set to 2 then the MGET subcommand should produce results similar to those below, which are identical to the GET of just the JOB00061.

At the end of each SYSOUT data set of an MGET, the following line is stored:

```
!! END OF JES SPOOL FILE !!
```

JESJOBName=* JESOwner=* JESSTATUS=ALL JESENTRYLimit=200 displays SYSOUT data sets from JOB00061.

```

DIR JOB00061
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER3A   JOB00061 USER3     OUTPUT  D 3 spool files RC=0000
          ID  STEPNAME PROCSTEP C DDNAME  REC-COUNT COMMENT
          001 JESE             H JESMSG LG      18
          002 JESE             H JESJCL  11
          003                 A JESYSMSG 22

MGET JOB00061.*
125 Sending all spool files for Jobid JOB00061
250 Transfer completed successfully.
5541 bytes transferred in 4.92 seconds. Transfer rate 1.35 Kbytes/sec.

```

If MGET is specified with an asterisk, it works the same as in JESINTERFACELevel 1. MGET gets all jobs that match the JES filters. Also, in the example below, there are two input jobs for user1, no active jobs, and three output jobs. Care should be taken with the JES filters because an MGET from the client actually appears to the server as an NLST followed by several GETs. For example, if JESSTATUS is set to ALL and there are many jobs returned in the NLST that are in ACTIVE or INPUT status, then the MGET will not retrieve the number of output files expected and there will be no message that "250 No tasks found for JESJOBName...". (JESJOBName=* JESOwner=USER1 JESSTATUS=ALL JESENTRYLimit=1024.)

```

DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A
USER1    JOB00022  USER1    ACTIVE  D STEP=STEPNAME PROC=PROCSTEP CPUT= 7.27 ELAPT= 7.27
USER1A   JOB00061  USER1    OUTPUT  D 3 spool files RC=0000
USER1B   JOB00070  USER1    OUTPUT  D 5 spool files RC=0000
USER1C   JOB00070  USER1    OUTPUT  D 17 spool files ABEND=0806
MGET JOB00061.*
125 Sending all spool files for Jobid JOB00061
250 Transfer completed successfully.
5541 bytes transferred in 4.92 seconds. Transfer rate 1.35 Kbytes/sec.

SITE JESENTRYLIMIT=2
200 Site command was accepted
DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1    TSU00017  USER1    INPUT   A
USER1    JOB00022  USER1    ACTIVE  D STEP=STEPNAME PROC=PROCSTEP CPUT= 7.27 ELAPT= 7.27
JESENTRYLIMIT of 2 reached. Additional entries not displayed.

```

Note that the DIR showed jobs TSU00017 and JOB00022, but the MGET is not able to retrieve them.

```

MGET
550 No spool files available for TSU00017
550 No spool files available for JOB00022
SITE JESSTATUS=OUTPUT
200 Site command was accepted
DIR
JOBNAME  JOBID    OWNER    STATUS  CLASS
USER1A   JOB00061  USER1    OUTPUT  D 3 spool files RC=0000
USER1B   JOB00070  USER1    OUTPUT  D 5 spool files RC=0000
MGET
125 Nlst started OK
250-JESENTRYLIMIT of 2 reached. Additional entries not received
250 Nlst completed successfully

125 Sending all spool files for requested Jobid JOB00061
250 Transfer completed successfully.
5541 bytes transferred in 0.394 seconds. Transfer rate 14.06 Kbytes/sec.

125 Sending all spool files for requested Jobid JOB00070
250 Transfer completed successfully.
5541 bytes transferred in 0.394 seconds. Transfer rate 14.06 Kbytes/sec.

```

Performing Db2 SQL queries with FTP

FTP enables you to submit a Structured Query Language (SQL) SELECT query to the Db2® subsystem and receive the results of the SQL query. FTP can perform this function as either the server or the client.

For information about installing the SQL query function for the FTP client or server, see the [z/OS Communications Server: IP Configuration Reference](#).

SQL data types supported by FTP

FTP access to SQL supports the following data types:

- DATE
- TIME
- TIMESTAMP
- VARCHAR (variable length, up to 254 characters)
- CHAR (fixed length, up to 254 characters)
- DECIMAL
- INTEGER (full word)
- SMALLINT (half word)
- FLOAT (single or double precision)
- LONG VARCHAR (VARCHAR(*n*), where *n* is greater than 254)

- GRAPHIC
- VARGRAPHIC
- LONG VARGRAPHIC

Mixed data (double-byte character set and single-byte character set) is supported in CHAR, VARCHAR, and LONG VARCHAR data types, but column alignment in the output file might not be maintained.

Creating the input data set

Before performing a Db2 SQL query using FTP, you must create an MVS data set that contains the SQL query you want to perform.

You can create queries on the client and use the FTP PUT command to send the queries to the MVS system to be processed. Or, you can prepare a group of SQL queries on the MVS system and perform them regularly.

Note: FTP can process only one SQL query per file.

For example, a data set on an MVS system named `userid.SQL.IN` contains the following SQL query:

```
SELECT LASTNAME, EMPID, YEARS_EMPLOYED FROM EMPLOYEE_TABLE
WHERE YEARS_EMPLOYED > 25
```

You either created that data set on the MVS system with TSO, or you used the FTP PUT command to put the data set on the MVS system.

Setting the characteristics for the SQL query

After creating a data set to use for your query, you must log on to FTP and set the file type for the query:

```
SITE/LOCSITE FILEtype=SQL
```

There are several commands that are relevant to the client and server in SQL mode. The server commands use SITE, and the client commands use LOCSITE. The following list describes the commands:

SITE/LOCSITE DB2®=

Specifies the name of the Db2 subsystem that you want to perform your queries. See [“Specifying the Db2 subsystem to perform the query” on page 132](#) for more information about Db2 subsystems.

SITE/LOCSITE SPRead or NOSPRead

Specifies whether you want the output to be in spreadsheet or report format. See [“Specifying the output format” on page 133](#) for more information about output format.

SITE/LOCSITE SQLCol=

Specifies whether you want the column headings to use the Db2 column names or labels. Valid values include Names, Labels, or Any. See [“Assigning column headings for the SQL query result table” on page 132](#) for more information about column headings.

To return to normal FTP processing after performing queries, or other processes, specify:

```
SITE/LOCSITE FILEtype=SEQ
```

On MVS systems, RECFM=VB is a recommended format that enables you to view the results of the SQL query. Issue the following command to specify that new data sets should be created with the RECFM=VB attribute:

```
SITE/LOCSITE RECFM=VB
```

To prevent the automatic sending of a SITE command that might override your SITE setting, toggle SENDSITE to OFF. For more information about the SENDSITE command, see [“SENDSite subcommand—Toggle the sending of site information” on page 246](#).

Specifying the Db2 subsystem to perform the query

An MVS system can run several Db2 systems simultaneously, each known by a subsystem name of up to four characters. For example, you can have a Db2 test system called DB2T and a Db2 production system called DB2P.

FTP connects to a Db2 system to have it execute a Db2 query. You can specify what Db2 system FTP should connect to with the following SITE or LOCSITE parameter:

```
SITE/LOCSITE DB2=
```

If you want the FTP server to have the DB2T system perform your queries, specify:

```
SITE DB2=DB2T
```

If you want the FTP client to have the DB2P system perform your queries, specify:

```
LOCSITE DB2=DB2P
```

The default Db2 system name is *DB2*. You can change the default with the DB2 parameter in the FTP.DATA data set. See [“Changing local site defaults using FTP.DATA”](#) on page 62 for more information about the FTP.DATA data set.

Assigning column headings for the SQL query result table

When you create a Db2 table, you can assign descriptive labels to the table columns. For example, a column name could be XCM554, but the label could be WEEKLY PAY. For information about assigning names and labels, see <http://www.ibm.com/support/knowledgecenter/SSEPH2/welcome>.

The SQLCol parameter of the SITE command enables you to specify whether you want names or labels to appear at the top of the columns in your output file. The default value is Names.

- Issue the following command if you want a database column name to appear at the top of each column in your output file:

```
SITE/LOCSITE SQLCol=Names
```

- Issue the following command if you want a label to appear at the top of each column:

```
SITE/LOCSITE SQLCol=Labels
```

If you specify the Labels parameter, and a column in your query does not have a label defined in the database, the FTP server supplies a column heading. For more information about column headings, see [“FTP-supplied column headings”](#) on page 132.

- Issue the following command if you want either a label or a name to appear at the top of each column:

```
SITE/LOCSITE SQLCol=Any
```

If you specify the Any parameter, the label appears as the column heading. However, if the column does not have a label, the name appears at the top of the column.

FTP-supplied column headings

The FTP client and server provide column headings in the result table when Db2 does not. This occurs when a result table contains expression columns or when labels are requested and a database column that appears in the result table does not have a label defined.

FTP builds a column heading for expression columns. For example,

```
Select employee, salary/52 from ABC.Staff
```

results in two columns. The first column gets its name from Db2, while the second column is built by the server. The server uses the heading COL002 for the second column because it supports the SQL limit of 750 columns.

Specifying the output format

You have two choices for the format of your output data set: spreadsheet format and report format. The default is NOSPRead (report format), but you can change the default for your FTP server by changing the FTP.DATA data set. See [“Changing local site defaults using FTP.DATA” on page 62](#) for more information.

Spreadsheet Format: You can have the output of the SQL query formatted to load directly into a spreadsheet program running on a PC or a workstation. To get the spreadsheet format, issue the following command:

```
SITE SPRead or LOCSITE SPRead
```

The SPRead format option puts a TAB character before the first character of each column entry, except the first column. See your spreadsheet program documentation for instructions about how to import the output of the SQL query.

Report Format: The NOSPRead format option puts one or more blank spaces between the columns, and it lists the SQL query, the column headings, and the resulting columns. Each section is separated with horizontal dashed lines. An output data set in NOSPRead, or report, format is easier to view and print.

To get the report format, issue the following command:

```
SITE NOSPRead or LOCSITE NOSPRead
```

The following is an example of the results contained in the NOSPRead format of the SQL.OUTPUT data set.

```
s-----+-----+-----+-----+-----+-----+-----+
SELECT * FROM DB2USER.PHONES
      WHERE FIRSTNAME LIKE 'BILL%'
      OR   FIRSTNAME LIKE 'WILL%'
h-----+-----+-----+-----+-----+-----+-----+
LASTNAME      FIRSTNAME      TIE  EXT  ALT  DEPT  ROOM  NODE
d-----+-----+-----+-----+-----+-----+-----+
ACKERMAN      BILL          893   6266  7813  431   J2-A22  IBMABC
ADAMS          WILLIAM J.    892   2202  1716  681   33-943  IBMABC
ASTERMAN      WILLIAM C.    893   7244  7813  222   J4-A44  IBMVM2
BENDER        WILLIAM R.    892   4217  4766  490   45-556  IBMVM2
```

A lowercase letter in the first position of each dashed line specifies what part of the output follows, enabling a program to read and interpret the contents. For example, s indicates that the SQL query follows, h indicates a header, d indicates that the rest of the data set is the actual data, and e indicates that an error message follows.

The width of the output data set depends on the width of the results from the Db2 query.

Submitting the query

After you have created a data set that contains an SQL query, logged on to FTP, and set the appropriate SITE or LOCSITE parameters, you are ready to execute the contents of the data set. You can do this from either an FTP client or an FTP server.

Performing an SQL query from an FTP client

Procedure

1. To have the FTP client perform SQL queries and have the results sent to an FTP server, specify:

```
LOCSITE FILEtype=SQL
```

2. Perform a PUT command specifying the name of the file on the client that contains the SQL query.

Example

For example, if the client has a file named `userid.SQL.IN` that contains an SQL query, specify:

```
PUT SQL.IN SQL.OUT
```

The FTP client then submits the query found in `SQL.IN` to the Db2 subsystem on the client and sends the resulting rows of output to the server to be put into `SQL.OUT` on the server.

To return to normal FTP processing, specify:

```
LOCSITE FILEtype=SEQ
```

Performing an SQL query from an FTP server

Procedure

1. To have the FTP server perform the query and have the results sent to the client, specify:

```
SITE FILEtype=SQL
```

2. Perform a GET command specifying the name of the file on the server that contains the SQL query.

Example

For example, if the server has a file named `userid.SQL.IN` that contains an SQL query, you can specify:

```
GET SQL.IN SQL.OUT
```

The FTP server then submits the query found in `SQL.IN` to the Db2 subsystem on the server and sends the resulting rows of output to the client to be put into `SQL.OUT` on the client.

Examples of SQL query output

This section shows examples of SQL query output using different options.

With NOSPRead and SQLCol=Names

The following output is from a query using `NOSPRead` and `SQLCol=Names`.

```
S-----+-----+-----+-----+-----+-----+-----+
SELECT EMPLOYEE,AGE
FROM   ABC.STAFF
WHERE  AGE < 60
h-----+-----+-----+-----+-----+-----+-----+
EMPLOYEE                                AGE
d-----+-----+-----+-----+-----+-----+-----+
Steve Jasinski                          23
Alison Cook                             22
```

With SPRead and SQLCol=Names

The following output is from a query with `SPRead` and `SQLCol=Names`.

Note: The period symbol (.) represents a TAB character.

```
EMPLOYEE      .AGE
Steve Jasinski . 23
Alison Cook    . 22
```

The following output examples are for the query:


```

SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)

```

With NOSPRead and SQLCol=Names

The following output is from queries using NOSPRead and SQLCol=Names.

```

s-----+-----+-----+-----+-----+-----+-----+
SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)
h-----+-----+-----+-----+-----+-----+-----+
TLA      SALARY
d-----+-----+-----+-----+-----+-----+-----+
ACO      20050.00
SJJ      19040.00

```

and

```

s-----+-----+-----+-----+-----+-----+-----+
SELECT * FROM ABC.STAFF
h-----+-----+-----+-----+-----+-----+-----+
EMPLOYEE          TLA      AGE      SALARY
d-----+-----+-----+-----+-----+-----+-----+
Steve Jasinski    SJJ      23      28040.00
Alison Cook       ACO      22      28040.00
Mark Ballam       MFB      63      87420.55

```

With NOSPRead and SQLCol=Labels

The following output is from a query using NOSPRead and SQLCol=Labels.

```

s-----+-----+-----+-----+-----+-----+-----+
SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)
h-----+-----+-----+-----+-----+-----+-----+
EMPLOYEE'S INITIALS  SALARY
d-----+-----+-----+-----+-----+-----+-----+
ACO                  20050.00
SJJ                  19040.00

```

With NOSPRead and SQLCol=Any

The following output is from a query using NOSPRead and SQLCol=Any.

```

s-----+-----+-----+-----+-----+-----+-----+
SELECT * FROM ABC.STAFF
h-----+-----+-----+-----+-----+-----+-----+
EMPLOYEE          EMPLOYEE'S INITIALS  AGE      SALARY
d-----+-----+-----+-----+-----+-----+-----+
Steve Jasinski    SJJ                  23      28040.00
Alison Cook       ACO                  22      28040.00
Mark Ballam       MFB                  63      87420.55

```

SUBSYS: Writing to BatchPipes

The FTP server supports binary transfer to IBM BatchPipes®. BatchPipes connect jobs so that data from one or more job can pass through processor storage to another job (or jobs) without being written to DASD or tape. For more information about BatchPipes see *IBM BatchPipes OS/390® Introduction*.

Steps for writing to BatchPipes

You can transfer a file or data set from the FTP client to the FTP server by writing to BatchPipes.

Procedure

To transfer a file to BatchPipes, perform the following steps:

1. Start the BatchPipes subsystem.

When you successfully start the BatchPipes subsystem, messages similar to the following ones are displayed on the MVS console.

```
11.33.43 IEF403I BP01 - STARTED - TIME=11.33.43
11.33.43 ASFP000I BATCHPIPES FOR OS/390 SUBSYSTEM BP01:
        PRODID=5655-D45 PRODLVL=HACH301 COMPID=565506500
        CONTAINS LICENSED MATERIALS - PROPERTY OF IBM CORP.
        CONTAINS RESTRICTED MATERIALS OF IBM CORP.
        5655-D45 (C) COPYRIGHT IBM CORP. 1992, 2000
        ALL RIGHTS RESERVED.
        U.S. GOVERNMENT USERS RESTRICTED RIGHTS -
        USE, DUPLICATION, OR DISCLOSURE RESTRICTED BY
        GSA ADP SCHEDULE CONTRACT WITH IBM CORP.
11.33.43 ASFP007I BATCHPIPES BP01 COMMAND PREFIX IS BP01
11.33.43 ASFP011I BATCHPIPES BP01 INITIALIZATION COMPLETE.
11.33.43 ASFP017I BATCHPIPES BP01 MODE(LOCAL) PIPEPLEX(**NONE**)
```

-
2. Start the BatchPipes reader.

You must start the BatchPipes reader before you can use the FTP server to write to BatchPipes. The following is an example of the Job Control Language (JCL) statements that you can use to start a BatchPipes subsystem reader of data set USER3.SUBSYS.OUTPUT3.

```
//USER302 JOB MSGCLASS=A,CLASS=A
//STEP1 EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=USER3.SUBSYS.OUTPUT2,SUBSYS=BP01,
//        DCB=(LRECL=80,RECFM=FB)
//SYSUT2 DD DSN=USER3.SUBSYS.OUTPUT3,
//        DISP=(NEW,CATLG,DELETE),
//        UNIT=SYSDA,
//        SPACE=(TRK,(10,10),RLSE),
//        DCB=(LRECL=80,RECFM=FB)
```

When BatchPipes is open for read, messages similar to the following are displayed in the JOBLOG output:

```
00 11.35.13 $HASP100 USER302 ON INTRDR FROM TSU00033 USER2
- 11.35.13 ICH70001I USER2 LAST ACCESS AT 11:05:39 ON FRIDAY, MARCH 2, 2007
- 11.35.13 $HASP373 USER302 STARTED - INIT 1 - CLASS A - SYS 3090
- 11.35.13 IEF403I USER302 - STARTED - TIME=11.35.13
- 11.35.14 ASFP394I BATCHPIPES READER JOB WAITING FOR OPEN.
- JOB=USER302 STEP=STEP1 DD=SYSUT1 SUBSYS=BP01
- PIPE=USER3.SUBSYS.OUTPUT2
```

-
3. Start the job that reads from BatchPipes.

Requirement: The BatchPipes reader must be active for the FTP server to write to BatchPipes.

-
4. Configure the FTP server for writing to BatchPipes.

Use the SITE command to specify the following:

- The BatchPipes subsystem name. Use the SUBSYS parameter. For example, if your BatchPipes subsystem is named BP01, specify the following:

```
SITE SUBSYS=BP01
```

- A record format that is compatible with BatchPipes. Supported record formats are F, FB, V, and VB. Specify the RECFM parameter.
- A logical record length and block size that are compatible with BatchPipes. Specify the LRECL and BLKSIZE parameters.

5. Set up the client and server for binary (Type I) file transfer.

Guideline: With most FTP clients, including the z/OS FTP client, you can use the Binary subcommand from the client to set up a binary file transfer.

Restrictions: The following restrictions apply when you specify a SUBSYS value:

- APPE and REST commands are not supported.
- Only binary (type I) file transfer is supported.
- Only FILETYPE SEQ is supported.
- Checkpointing and file transfer restart are not supported. Checkpointing is described in [“Restarting a failed data transfer”](#) on page 103.
- SMS-managed data sets (data sets with an assigned storage class) cannot be used.
- Only RECFM values F, FB, V, and VB are supported.

6. Transfer a file or data set to the FTP server, specifying BatchPipes as the destination.

You must put directly to the BatchPipes subsystem rather than appending to BatchPipes, for example:

```
put 'user3.source.data' 'user3.subsys.output1'
```

SUBSYS examples

The following are examples of using FTP to transfer a file to IBM BatchPipes.

Example 1:

In this example, the FTP server writes the file USER3.SUBSYS.OUTPUT1 to a BatchPipes reader. This JCL starts the BatchPipes reader. The file USER3.SUBSYS.OUTPUT1 has the record format FB and a logical record length of 80. The name of the BatchPipes subsystem is BP01 as specified by the SUBSYS parameter.

```
//USER3@2 JOB MSGCLASS=A,CLASS=A
//STEP1 EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=USER3.SUBSYS.OUTPUT1,SUBSYS=BP01,
//          DCB=(LRECL=80,RECFM=FB)
//SYSUT2 DD DSN=USER3.SUBSYS.OUTPUT2,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(10,10),RLSE),
//          DCB=(LRECL=80,RECFM=FB)
```

Because the BatchPipes reader has the logical record length 80 and a record format FB, you must configure the FTP server to use those values for the file transfer:

```
230 USER1 is logged on. Working directory is "USER1.".
Command:
ftp> bin                               ====> Setting Binary Transfer
200 Representation type is Image
```

```

Command:
site lrecl=80 recfm=fb subsys=bp01          =====> Sets SITE variables for
                                           =====> RECFM, LRECL, and SUBSYS.

>>> SITE lrecl=80 recfm=fb subsys=bp01
200-BLOCKSIZE must be a multiple of LRECL for RECFM FB
200-BLOCKSIZE being set to 6160
200 SITE command was accepted
Command:
put 'user3.source.data' 'user3.subsys.output1' =====> Transferring
'user3.source.data'
                                           =====> to Reader end of the

BatchPipes
                                           =====> 'user3.subsys.output1'

>>> PORT 9,42,104,22,4,6
200 Port request OK.
>>> STOR 'user3.subsys.output1'
125-Waiting for Batchpipes subsystem BP01 reader end to open.
125 Storing data set USER3.SUBSYS.OUTPUT1
250 Transfer completed successfully.
820 bytes transferred in 0.005 seconds. Transfer rate 164.00 Kbytes/sec.
Command:

```

Example 2:

```

//USER3@2 JOB MSGCLASS=A,CLASS=A
//STEP1 EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=USER3.SUBSYS.VBOUPT1,SUBSYS=BP01,
//          DCB=(LRECL=32756,RECFM=VB)
//SYSUT2 DD DSN=USER3.SUBSYS.VBOUPT2,
//          DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,
//          SPACE=(TRK,(10,10),RLSE),
//          DCB=(LRECL=32756,RECFM=VB)

```

Because the BatchPipes reader has the logical record length 32 756 and the record format VB, you must configure the FTP server to use those values for the file transfer. For this example use record structure.

```

ftp> bin          =====> Setting Binary Transfer
200 Representation type is Image
ftp> quote stru r  =====> Setting Record Structure
250 Data structure is Record
ftp> quote site lrecl=32756 recfm=vb subsys=bp01 =====> Setting SITE variables for
                                           =====> RECFM, LRECL, and SUBSYS.
200-BLOCKSIZE must be at least 4 more than LRECL for RECFM VB
200-BLOCKSIZE being set to 32760
200 SITE command was accepted
ftp> put new.txt 'user3.subsys.vboupt1'
200 Port request OK.
125-Waiting for Batchpipes subsystem BP01 reader end to open.
125 Storing data set USER3.SUBSYS.VBOUPT1

250 Transfer completed successfully.
ftp: 126 bytes sent in 0.00Seconds 126000.00Kbytes/sec .
ftp>

```

For more information, see the following:

- [z/OS MVS JCL Reference](#)
- [IBM BatchPipes OS/390 Introduction](#)
- [SmartBatch BatchPipeWorks Reference](#)
- [z/OS MVS Programming: Authorized Assembler Services Guide](#)

Chapter 5. FTP subcommands

The FTP subcommands are listed in [Table 20 on page 139](#). The minimum abbreviation, a description, and the link to each subcommand are also included. You must be in the FTP environment to use the FTP subcommands. See [“Using FTP” on page 19](#) for more information.

Table 20. FTP subcommands

| Subcommand | Description | See |
|------------|--|---|
| ? | Provides information to use FTP. | “HElp and ? subcommands—Display help information” on page 171 |
| ! | Passes a z/OS UNIX System Services command to the local z/OS shell. This command must be issued while using FTP in the z/OS shell. | “! subcommand—Invoke a z/OS UNIX System Services function” on page 143 |
| ACCount | Sends host-dependent account information. | “ACcT subcommand—Supply account information ” on page 144 |
| APpend | Appends a data set on your local host to a file on the foreign host. | “APpend subcommand—Append a local data set ” on page 144 |
| AUth | Requests a security mechanism for the session. | “AUth subcommand—Request security mechanism” on page 146 |
| AScii | Sets the transfer type to ASCII. | “AScii subcommand—Change the data transfer type to ASCII ” on page 146 |
| BIG5 | Sets the transfer type to BIG5. BIG is the minimum abbreviation for BIG5. | “BIG5 subcommand—Change the data transfer type to BIG5 ” on page 146 |
| BINary | Sets the transfer type to IMAGE. | “BINary subcommand—Change the data transfer type to Image ” on page 147 |
| BLock | Sets the data transfer mode to block mode. This is equivalent to specifying the MObd B subcommand. | “BLocK subcommand—Set the block data transfer mode” on page 148 |
| CCc | Turns off integrity protection on the command channel. | “CCc subcommand—Turn off integrity protection” on page 148 |
| CD | Changes the working directory. | “CD subcommand—Change the directory on the remote host ” on page 149 |
| CDUp | Changes to the parent of the current working directory. | “CDUp subcommand—Change to the parent of the working directory” on page 151 |
| CLEar | Sets the protection level for data transfers to CLEAR. | “CLEar subcommand—Set the protection level for data transfers to CLEAR” on page 152 |
| CLOSE | Disconnects from the foreign host. | “CLose subcommand—Disconnect from a remote host ” on page 153 |
| COMpress | Sets the data transfer mode to compressed mode. This is equivalent to specifying the MObd C subcommand. | “COMpress subcommand—Set the compressed data transfer mode” on page 153 |

Table 20. FTP subcommands (continued)

| Subcommand | Description | See |
|------------|---|---|
| CProtect | Sets the protection level on commands. CProtect CLEAr is equivalent to the CCc command. | “CProtect subcommand— Set the protection level on commands” on page 154 |
| CWd | Changes the working directory (Synonymous with CD). | “CD subcommand—Change the directory on the remote host ” on page 149 |
| DEBUg | Enables or disables general internal tracing. | “DEBUg subcommand—Set general trace options” on page 154 |
| DELEte | Deletes a single file on the foreign host. | “DELEte subcommand—Delete files ” on page 158 |
| DELImit | Displays the delimiter character between the <i>file_name</i> and <i>file_type</i> . | “DELImit subcommand—Display the file name delimiter ” on page 158 |
| DIr | Lists the directory entries for files on the foreign host. | “DIr subcommand—Obtain a list of directory entries ” on page 159 |
| DUMP | Enables or disables extended internal tracing. | “DUMP subcommand—Set extended trace options” on page 163 |
| EBcdic | Sets the transfer type to EBCDIC. | “EBcdic subcommand—Change the data transfer type to EBCDIC ” on page 164 |
| EUckanji | Sets the transfer type to EUCKANJI. | “EUckanji subcommand—Change the data transfer type to EUCKANJI” on page 165 |
| FEature | Queries FTP Server for features it supports. | “FEature subcommand—Query FTP server for features it supports” on page 166 |
| File | Sets the file structure to file. This is equivalent to specifying the STRucture F subcommand. | “File subcommand—Set the file structure to File” on page 166 |
| Get | Copies a file from the foreign host to your local host. | “Get subcommand—Copy files ” on page 166 |
| GLob | Toggles globbing (the expansion of metacharacters in file names) for the MDelete, MGet, and MPut subcommands. | “GLob subcommand—Toggle expansion of metacharacters” on page 168 |
| HAngeul | Sets the transfer type to HANGEUL. | “HAngeul subcommand—Change the data transfer type to HANGEUL ” on page 170 |
| HElp | Displays help information for FTP. | “HElp and ? subcommands—Display help information” on page 171 |
| Ibmkanji | Sets the transfer type to IBMKANJI. | “Ibmkanji subcommand—Change the data transfer type to IBMKANJI” on page 171 |
| JIS78kj | Sets the transfer type to JIS78KJ. | “JIS78kj subcommand—Change the data transfer type to JIS78KJ ” on page 172 |
| JIS83kj | Sets the transfer type to JIS83KJ. | “JIS83kj subcommand—Change the data transfer type to JIS83KJ ” on page 173 |
| Ksc5601 | Sets the transfer type to KSC5601. | “Ksc5601 subcommand—Change the data transfer type to KSC-5601 ” on page 173 |

Table 20. FTP subcommands (continued)

| Subcommand | Description | See |
|------------|---|--|
| LANGuage | Sets the language used for FTP replies from the server. | “LANGuage subcommand—Set the language used for FTP replies from the server” on page 174 |
| LCd | Changes the current directory on the local host. | “LCd subcommand—Change the local working directory ” on page 175 |
| LOCSite | Specifies information that is used by the local host to provide service specific to that host system. | “LOCSite subcommand—Specify site information to the local host” on page 179 |
| LOCStat | Displays FTP status information for the local host. | “LOCStat subcommand—Display local status information ” on page 201 |
| LMkdir | Creates a directory on the local host. | “LMkdir subcommand—Create a directory on the local host ” on page 177 |
| LPwd | Displays the name of the active working directory on the local host. | “LPwd subcommand—Display the current working-level qualifier ” on page 208 |
| LS | Lists the names of files on the foreign host. | “LS subcommand—Obtain a list of file names ” on page 208 |
| MDelete | Deletes multiple files on the foreign host. | “MDelete subcommand—Delete multiple files ” on page 211 |
| MGet | Copies multiple files from the foreign host to your local host. | “MGet subcommand—Copy multiple files ” on page 213 |
| MKdir | Creates a directory on the foreign host. | “MKdir subcommand—Create a directory on the remote host ” on page 216 |
| MKFifo | Creates a UNIX named pipe on the remote host. | “MKFifo subcommand—Create a named pipe at the FTP server host” on page 219 |
| MOfde | Specifies the mode or data format of the transfer. | “MOfde subcommand—Set the data transfer mode ” on page 220 |
| MPut | Copies multiple files on your local host to the foreign host. | “MPut subcommand—Copy multiple data sets to the remote host ” on page 221 |
| MVSGet | Copies a remote data set into a local data set with the remote data set attributes | “MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes” on page 223 |
| MVSPut | Copies a local data set into a remote data set with the local data set attributes | “MVSPut subcommand – Copy a local data set into a remote data set name with the local data set attributes” on page 227 |
| NOop | Checks whether the foreign host is still responding. | “NOop subcommand—Test the connection ” on page 230 |
| Open | Opens a connection to a foreign host. | “Open subcommand—Connect to the FTP server ” on page 230 |
| PAss | Supplies a password or password phrase to the foreign host. | “PAss subcommand—Supply a password ” on page 231 |
| PRivate | Sets the protection level for data transfers to PRIVATE. | “PRivate subcommand—Set the protection level for data transfers to PRIVATE” on page 233 |

Table 20. FTP subcommands (continued)

| Subcommand | Description | See |
|------------|---|--|
| PROMpt | Toggles interactive prompting for MDelete, MGet, and MPut commands. This function is similar to specifying the FTP command with the -i option, which turns off interactive prompting. | “PROMpt subcommand—Toggle interactive prompting for M* commands” on page 234 |
| PROtect | Sets the protection level for data transfers on the data connections. | “PROtect subcommand—Set the protection level for data transfers” on page 234 |
| PROxy | Executes an FTP subcommand on a secondary control connection. | “PROxy subcommand—Execute FTP subcommand on secondary control connections” on page 235 |
| PUt | Copies a file on your local host to the foreign host. | “PUt subcommand—Copy data sets to the remote host ” on page 238 |
| PWd | Displays the name of the active working directory on the foreign host. | “PWd subcommand—Display the current working directory ” on page 239 |
| QUIt | Leaves the FTP command environment. | “QUIt subcommand—Leave the FTP environment ” on page 240 |
| QUOte | Sends an uninterpreted string of data. | “QUOte subcommand—Send an uninterpreted string of data ” on page 240 |
| RECORD | Sets the file structure to record. This is equivalent to specifying the STRucture R subcommand. | “RECORD subcommand—Set the file structure to record” on page 241 |
| REName | Renames a file on the foreign host. | “REName subcommand—Rename files ” on page 242 |
| REstart | Restarts a checkpointed data transfer. | “REstart subcommand - Restart a checkpointed data transfer” on page 242 |
| RMdir | Removes a directory. | “RMdir subcommand—Remove a directory on the remote host” on page 244 |
| SAfe | Sets the protection level on data transfers to <i>safe</i> . | “SAfe subcommand—Set the protection level to safe” on page 244 |
| SCHinese | Sets the transfer type to SCHINESE. | “SCHinese subcommand—Change the data transfer type to SCHINESE” on page 244 |
| SENDPort | Enables or disables automatic transmission of the FTP server PORT command. | “SENDPort subcommand—Toggle the sending of port information ” on page 245 |
| SENDSite | Enables or disables automatic transmission of the SItE subcommand. | “SENDSite subcommand—Toggle the sending of site information ” on page 246 |
| SItE | Sends information to the foreign host using site-specific commands. | “SItE subcommand—Send site-specific information to a host ” on page 247 |
| SJiskanji | Sets the transfer type to SJISKANJI. | “SJiskanji subcommand—Change the data transfer type to SJISKANJI” on page 275 |
| SRestart | Restarts an interrupted stream mode data transfer. | “SRestart subcommand—Restart a stream data transfer” on page 276 |

Table 20. FTP subcommands (continued)

| Subcommand | Description | See |
|------------|--|---|
| STatus | Displays status information for the foreign host. | “STatus subcommand—Retrieve status information from a remote host” on page 278 |
| STREam | Sets the data transfer mode to stream mode. This is equivalent to specifying the MDe S subcommand. | “STREam subcommand—Set the stream data transfer mode” on page 285 |
| STRucture | Sets the file transfer structure. | “STRucture subcommand—Set the file structure ” on page 286 |
| SUnique | Toggles the storage methods. | “SUnique subcommand—Changes the storage method ” on page 286 |
| SYstem | Displays the name of the foreign host operating system. | “SYstem subcommand—Display the operating system name ” on page 287 |
| TChinese | Sets the transfer type to TCHINESE. | “TChinese subcommand—Change the data transfer type to TCHINESE” on page 287 |
| TSO | Passes a TSO command to the local host TSO environment. | “TSO subcommand—Use TSO commands ” on page 288 |
| TyPe | Specifies the transfer type. | “TyPe subcommand—Set the data transfer type ” on page 289 |
| UCs2 | Changes the data transfer type to Unicode UCS-2. UC is the minimum abbreviation for UCs2. | “UCs2 subcommand—Change data transfer type to Unicode UCS-2” on page 293 |
| User | Identifies you to a foreign host or changes your TSO user ID password or password phrase. | “User subcommand—Identify yourself to a host or change your TSO user ID password” on page 293 |
| Verbose | Enables or disables verbose mode. | “Verbose subcommand - Toggle verbose mode” on page 296 |

! subcommand—Invoke a z/OS UNIX System Services function

Purpose

In a z/OS UNIX environment, use the ! subcommand to invoke z/OS UNIX functions.

Format

```
➔ ! _____ ➔
    |
    | shell_command
    |
```

Parameters

shell_command

Specifying the ! subcommand with a shell command enables you to invoke z/OS UNIX, perform the subcommand, and return to the FTP environment.

If no shell command is specified, the ! subcommand invokes z/OS UNIX. There you can specify any number of shell commands before typing `exit` to return to the FTP environment.

ACcT subcommand—Supply account information

Purpose

Use the ACcT subcommand to supply account information to a host.

Format

➤ ACcT — *account_information* ➤

Parameters

account_information

Specifies the account information required by the host. See your foreign-host FTP server documentation for the information required by that host.

Usage

- The z/OS FTP server does not require any account information.
- You might have to use the ACcT subcommand when the foreign host requires passwords for read and write access to its files or data sets. If you are not prompted by the foreign host for the passwords, use the ACcT subcommand to send these passwords to the foreign host.

APpend subcommand—Append a local data set

Purpose

Use the APpend subcommand to append a local data set to a remote host.

Format

➤ APpend — *local_data_set* — *destination_file* ➤

Parameters

local_data_set

The name of the data set on your local host to be appended.

destination_file

The name of the file on the remote host to which your data set is appended. If the destination file does not already exist at the remote host, a new file is created. If the server is a z/OS UNIX server, the local file can be appended to a z/OS UNIX file or an MVS data set.

Examples

In the following example, an FTP command is issued from MVSXA2 to MVSXA3. MVSXA2 has a data set MVSUSER.FTP.EXAMPLE with one member. The member, APPEND01, contains:

```
;
; THIS FILE ORIGINALLY RESIDED IN MVSXA2, AND
; WILL BE APPENDED TO A FOREIGN FILE IN MVSXA3.
;
```

MVSXA3 has a data set, MVSUSER.FTP.EXAMPLE, with one member, APPEND02. The member contains:

```
;
; THIS FILE ORIGINALLY RESIDED IN MVSXA3, AND
; WILL BE USED TO RECEIVE ANOTHER FILE FROM MVSXA2.
;
```

```
User: append
System: Usage: APPEND localfile foreignfile
        Command:

User: lpwd
System: Local directory is MVSUSER.
        Command:

User: append 'mvsuser.ftp.example(append01)' 'mvsuser.ftp.example(append02)'
System: >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
        200 Site command was accepted
        >>>PORT 1,1,2,2,4,16
        200 Port request OK.
        >>>APPE 'mvsuser.ftp.example(append02)'
        125 Appending to data set MVSUSER.FTP.EXAMPLE(APPEND02)
        250 Transfer completed successfully.
        520 bytes transferred in 1.100 seconds.
        Transfer rate 0.47 Kbytes/sec.
        Command:
```

Results:

- The following information applies when the *local_data_set* value is a named pipe in the z/OS UNIX file system:
 - FTP cannot send the named pipe until you start a process on the client host to write to the named pipe. If FTP is the first process to open the named pipe, it blocks until another process opens the named pipe for writing, or until the FIFOPENTIME timer expires.
 - Appending a named pipe to a remote file permanently removes data from the named pipe in the FTP client file system.
- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the APPEND subcommand, FTP can truncate data records and you might lose data. If the data set name already exists at the receiving site and the logical record length (LRECL) of the data set at the receiving site is less than the LRECL of the transmitted data set, FTP truncates the transmitted data set.
- If the remote host is an MVS or VM host, and if the data set on the remote host has a fixed-record format, the format and record length of the data set on the remote host are always preserved.
- Records from the data set on your local host are truncated or padded with blank spaces when necessary.
- To append to a file on a remote host, you must define a working directory on that host, and you must have write privileges to the files in that directory.
- The z/OS FTP Server does not request that unused space be released from a data set created during APPEND processing. If you are using the z/OS FTP Server and want the Server to request that unused space be released on a newly-created data set, use the PUT subcommand instead of APPEND.

Related topics:

- See [“Using z/OS UNIX System Services named pipes” on page 104](#) for more information about using named pipes.
- See [“CD subcommand—Change the directory on the remote host ” on page 149](#) for more information about working with current directories.
- See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about naming conventions.
- APPEND can be used with the PROXY subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See [“PROXY subcommand—Execute FTP subcommand on secondary control connections” on page 235](#) for more information.

AScii subcommand—Change the data transfer type to ASCII

Purpose

Use the AScii subcommand to change the data transfer type to ASCII.

Format

➤ AScii ➤

Parameters

There are no parameters for this subcommand.

Usage

Use the ASCII subcommand to direct FTP to translate outgoing files into ASCII before sending them to the other host, and to convert incoming files from ASCII to the file system code page before storing them.

Context

For more information about transfer methods, see [Table 13 on page 46](#).

AUth subcommand—Request security mechanism

Purpose

Use the AUth subcommand to request a security mechanism for the session.

Format

➤ AUth — *security_mechanism* ➤

Parameters

security_mechanism

The possible value is:

TLS

Request or reset TLS security for the session.

Result: The server might not support the security mechanism that you specify, or it might not accept the security mechanism that you specify.

Restrictions:

- This subcommand is not valid with a TLS_{PORT} implicit connection.
- This subcommand is not valid during a TLS-secured session when TLSRFCLEVEL parameter is set to DRAFT. See [“Using security mechanisms” on page 39](#) for more information.
- This subcommand is not valid during a Kerberos-secured session.

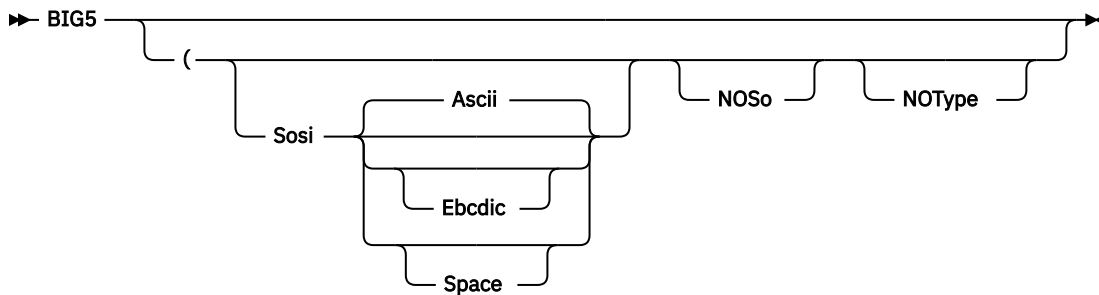
BIG5 subcommand—Change the data transfer type to BIG5

Purpose

Use the BIG5 subcommand to change the data transfer type to BIG5.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the `Site/LOCSite SBDataconn` or `Site XLate` subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters – `Ascii`, `Ebcdic`, or `Space`. If no parameter is specified, ASCII is used as the default.

If `Sosi` is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the `Sosi` parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the `Sosi` parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the `Sosi` parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to and from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding `TType` command to the server. Use this parameter when translation is to be performed by the FTP client only.

Usage

- The `BIG5` client subcommand is equivalent to the `TYPE B 8` command.
- The minimum abbreviation for `BIG5` is `BIG`.

Context

See [“FTP with traditional DBCS support” on page 78](#) and [“Support for MBCS languages” on page 81](#) for more information.

BINary subcommand—Change the data transfer type to Image

Purpose

Use the `BINary` subcommand to change the data transfer type to image (binary).

Format

► BINary ◀

Parameters

There are no parameters for this subcommand.

Usage

Use the image transfer type to transfer files between client and server without any translation of the file data. When using the image transfer type, data is sent as contiguous bits packed into 8-bit bytes. Use the image transfer type for efficient storage and retrieval of data sets or files, and for the transfer of binary data.

Context

For more information about data transfer methods, see [Table 13 on page 46](#).

BBlock subcommand—Set the block data transfer mode

Purpose

Use the BBlock subcommand to set the data transfer mode to block mode. This is equivalent to specifying the MOb B subcommand. See [“MOde subcommand—Set the data transfer mode” on page 220](#) for more information.

Format

► BLock ◀

Parameters

There are no parameters for this subcommand.

CCc subcommand—Turn off integrity protection

Purpose

Use the CCc subcommand to turn off integrity protection on the control connection. This command must be integrity-protected, and must be preceded by a successful security mechanism negotiation.

Format

► CCc ◀

Parameters

There are no parameters for this subcommand.

Rule: Because turning off integrity protection potentially enables an attacker to insert commands onto the control connection, some FTP servers might refuse to honor this command.

Restrictions: When the security mechanism is TLS, the following restrictions apply:

- The CCc subcommand is not supported when the connection is implicitly secured with a connection to the port that is configured with the TLS`PORT` statement.
- The CCc subcommand is supported only when the TLSRFCLEVEL is RFC4217 or CCCNONOTIFY.

CD subcommand—Change the directory on the remote host

Purpose

Use the CD subcommand to change the working directory or file group on the remote host.

Format

➤ CD — *directory* ➤

Parameters

directory

Specifies the name of a file directory, a fully qualified data set, or a prefix on the remote host.

Examples

Usage

You can also use the CWD and CW subcommands to change the current working directory. These subcommands are synonyms of the CD subcommand.

Changing the directory of a z/OS FTP server

If the remote server is z/OS FTP, the *directory* value can specify either a z/OS UNIX file system name, a common prefix for a group of MVS data sets, or the qualifiers of a partitioned data set (PDS).

Procedure

- When the CD subcommand is issued, the *directory* specified is appended to the current working directory.

For example, if the current working directory is TCPUSR14.TEST, and you issue the CD subcommand:

```
CD FILES
```

the new working directory becomes TCPUSR14.TEST.FILES.

- To override the existing directory rather than append to the directory, issue the *directory* parameter within single quotation marks.

For example, if the current working directory is TCPUSR14.TEST, and you issued the CD subcommand:

```
CD 'FTP.FILES'
```

the new working directory would be FTP.FILES. If the subdirectory name contains white space, such as NEW SUBDIRECTORY, then the syntax for the CD command would be as follows:

```
CD 'NEW SUBDIRECTORY'
```

The command syntax must specify the full subdirectory name (including the blank) delimited within single quotation marks.

- If a PDS exists with the exact name of the current working directory, FTP considers the working directory to be that PDS. Otherwise, FTP considers the working directory to be a common prefix qualifier for sequential data sets.

- If a PDS exists with the same name as the current working directory, but you want the current working directory to be treated as a common prefix for sequential data sets, specify the working directory with a period (.) at the end.

For example, if a PDS named TCPUSR14.TEST exists, the subcommand:

```
CD 'TCPUSR14.TEST'
```

makes the PDS TCPUSR14.TEST the current working directory. A subsequent PUt of file name1 adds a member name1 to the TCPUSR14.TEST PDS. In contrast, the subcommand

```
CD 'TCPUSR14.TEST.'
```

makes the current working directory, TCPUSR14.TEST., a prefix for sequential data sets. A subsequent PUt command used to copy data set name1 would create the sequential data set TCPUSR14.TEST.name1.

- To back up one level of the current working directory, issue the CD subcommand with two periods (..) at the end.

For example, if the working directory is jones.source, the subcommand

```
CD ..
```

makes jones. the working directory. You can also use the CDUp command to back up one level of the current working directory. See [“CDUp subcommand—Change to the parent of the working directory”](#) on page 151 for more details.

Example

The following sample commands and responses are displayed as a result of the CD subcommand.

For an MVS data set:

```
cd hsmtest
>>>CWD hsmtest
250 "'USER17.HSMTEST.'" is working directory name prefix.
Command:
```

For a z/OS UNIX file:

```
cd '/u/user121/A/B/C'
>>>CWD '/u/user121/A/B/C'
250 HFS directory /u/user121/A/B/C is the current working directory
Command:
```

Changing the directory of a VM FTP server

If the remote host is using TCP/IP for VM, the directory can be specified in either of the following ways:

- *user_id minidisk_address*
- *user_id.minidisk_address*

For example, to access the 191 minidisk of user ID jones, enter one of the following command:

- jones 191
- jones.191

Testing throughput with *DEV.NULL

If you have a z/OS FTP server, you can use the PUT or MVSPut subcommand to copy many files (or one large file) without storing the files on a z/OS FTP server file system. This is useful for testing purposes because you do not have to worry about allocating the disk space on the server system.

Procedure

1. Change the working directory to *DEV.NULL by using one of the following commands:

- CD *DEV.NULL
- CWD *DEV.NULL

This affects the working directory for only the PUT or MVSPut subcommand.

2. Use the PUT or MVSPut subcommand to copy the file to the server system. The input data set must be valid, and the output file can either be new or already exist. In either case, the file is not actually stored. The following response shows information such as the number of bytes transferred and the rate of transfer.
3. To end the use of the *dev.null directory for the PUT commands or MVSPut subcommand, issue another change directory command.

Example

```
Command:
cd *dev.null
>>>CWD *dev.null
250-Working directory for PUT is NULL Device;
250 for GET is HFS directory /u/user31
Command:
put a.b a.bbbbb
>>>SITE VARrecfm Lrecl=128 Recfm=VB BlockSize=6144
200 Site command was accepted
>>>PORT 14,0,0,0,4,14
200 Port request OK.
>>>STOR a.bbbbb
125 Storing data set in the Null directory (*dev.null).
250 Transfer completed successfully.
82 bytes transferred in 0.245 seconds. Transfer rate 0.33 Kbytes/sec.
Command:
quit
>>>QUIT
221 Quit command received. Goodbye.
```

CDUp subcommand—Change to the parent of the working directory

Purpose

Use the CDUp subcommand as a special case of the CD subcommand to change the working directory to the next higher directory level. You can use it to simplify the implementation of programs for transferring directory trees between operating systems that have different syntaxes for naming the parent directory.

Format

➡ CDUp →

Parameters

There are no parameters for this subcommand.

Examples

Change the working directory to the next higher directory level:

```

cd 'a.b.c.d'
>>>CWD 'a.b.c.d'
257 "'A.B.C.D.'" is working directory name prefix.
Command: pwd

>>>PWD
257 "'A.B.C.D.'" is working directory
Command: cdup

>>>CDUP
257 "'A.B.C.'" is working directory name prefix.
Command: pwd

>>>PWD
257 "'A.B.C.'" is working directory
Command: cdup

>>>CDUP
257 "'A.B.'" is working directory name prefix.
Command: pwd

>>>PWD
257 "'A.B.'" is working directory

```

Change the working directory to the next higher directory level for a z/OS UNIX file:

```

cd '/u/user121/A/B/C'
>>>CWD '/u/user121/A/B/C'
250 HFS directory /u/user121/A/B/C is the current working directory
Command: pwd

>>>PWD
257 "/u/user121/A/B/C" is the HFS working directory
Command: cdup

>>>CDUP
250 HFS directory /u/user121/A/B is the current working directory
Command: pwd

>>>PWD
257 "/u/user121/A/B" is the HFS working directory
Command: cdup

>>>CDUP
250 HFS directory /u/user121/A is the current working directory
Command: pwd

>>>PWD
257 "/u/user121/A" is the HFS working directory.
Command:

```

CLear subcommand—Set the protection level for data transfers to CLEAR

Purpose

Use the CLear subcommand to set the protection level for data transfers on the data connections to clear. This subcommand is equivalent to specifying the PROtect CLear subcommand.

Format

➤ CLear ➤

Parameters

There are no parameters for this subcommand.

Examples

To set the protection level to clear, enter:

```
clear
```

Usage

See the “[PROtect subcommand—Set the protection level for data transfers](#)” on page 234 for additional protection level information.

The CLEar subcommand is not valid when there is no active security mechanism.

Close subcommand—Disconnect from a remote host

Purpose

Use the CClose subcommand to disconnect from the remote host and remain in FTP.

Format

➤ CClose ➤

Parameters

There are no parameters for this subcommand.

Usage

The FTP session remains active on your local host, but the session to the remote host is terminated. You can use the Open subcommand to establish a new session with either the same or a different remote host. If you establish a new session with the same remote host, values set by the SItE subcommand during the previous session are cleared. The remote host default values for the parameters of the SItE subcommand are used for the new session.

Context

- See “[Open subcommand—Connect to the FTP server](#)” on page 230 for information about the Open subcommand.
- CClose can be used with the PROXy subcommand to close a secondary control connection. See “[PROXy subcommand—Execute FTP subcommand on secondary control connections](#)” on page 235 for more information.

COMpress subcommand—Set the compressed data transfer mode

Purpose

Use the COMpress subcommand to set the data transfer mode to compressed mode. This is equivalent to specifying the MOdE C subcommand. See “[MOdE C subcommand—Set the data transfer mode](#)” on page 220 for more information.

Format

➤ COMpress ➤

Parameters

There are no parameters for this subcommand.

CProtect subcommand— Set the protection level on commands

Purpose

Set the protection level on commands to *protection-level*.

Format

➤ CProtect — [*protection-level*] ➤

Parameters

The valid protection levels are:

protection-level

Can have the following values:

clear

Unprotected commands

safe

Commands integrity-protected by cryptographic checksum

private

Commands confidentiality and integrity-protected by encryption

Result: If an ADAT command succeeds, the default command protection level is *safe*; otherwise, the only possible level is *clear*. If no level is specified, the current level is used.

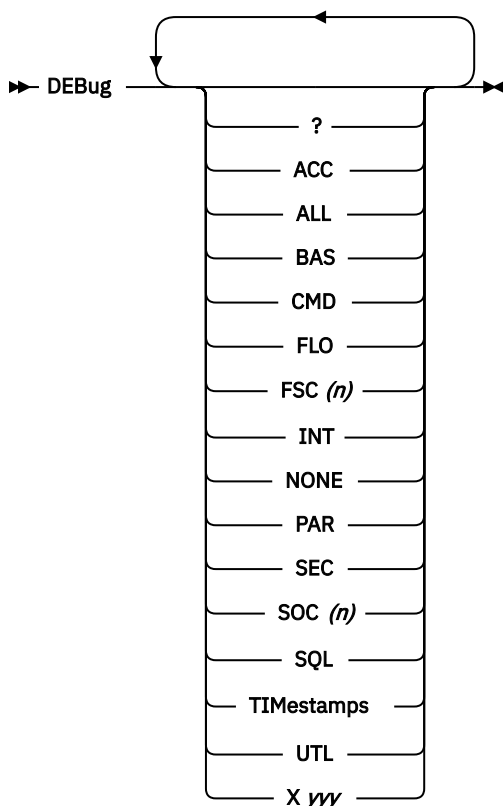
Tip: CProtect CLEAr is equivalent to the CCc subcommand.

DEBug subcommand—Set general trace options

Purpose

Use the DEBug subcommand to enable or disable general internal tracing.

Format



Parameters

One or more of the following options can be specified:

?

Displays the status of the traces.

ACC

The ACC trace shows the details of the login process.

ALL

This parameter is used to set all of the trace points.

Note: Both the FSC and the SOC traces will be set to level 1 when the ALL parameter is processed.

BAS

This parameter is used to set a select group of traces that offer the best overall debug data without the detailed information of some of the traces. Specifying this parameter is the same as `DEBUG CMD INT FSC SOC`.

CMD

The CMD trace shows each command and the parsing of the parameters for the command.

FLO

The FLO trace shows the flow of control within FTP. It is useful to show which services of FTP are used for an FTP request.

FSC(*n*)

The FSC trace shows details of the processing of the file services subcommands `APpend`, `PUt`, `Get`, `MGet`, and `MPut`. This trace can generate very detailed information and therefore allows you to specify levels of detail for the trace points.

The level 1 tracing that is specified by entering `FSC` or `FSC(1)` is the level normally used unless more data is requested by TCP/IP service group. The level (*n*) can be a number from 1 to 8.

INT

The INT trace shows the details of the initialization and termination of the FTP session with the server.

NONE

This parameter is used to turn off all of the traces.

PAR

The PAR trace shows details of the FTP command parser. It is useful for debugging problems in the handling of the command parameters.

SEC

The SEC trace shows the processing of security functions such as TLS and GSSAPI negotiations.

SOC(*n*)

The SOC trace shows details of the processing during the setup of the interface between the FTP application and the network as well as details of the actual amounts of data that is processed.

This trace can generate very detailed information and therefore allows you to specify levels of detail for the trace points. The level (*n*) can be a number from 1 to 8.

SQL

The SQL trace shows details of the processing for SQL requests, such as requests when LOCSite FILETYPE=SQL is in effect.

TIMestamps

This is a special parameter used to request that each trace entry have a timestamp showing the time the entry was made. This is useful during long running file transfers to track the duration of the transfer and also to match client trace to the server trace, which also has a timestamp.

UTL

The UTL trace shows the processing of utility functions such as LCd and LOCSite.

Xyyy

This syntax is used to turn off (reset) a trace that is named by yyy. For example, DEBUG XPAR XACC will turn off the PAR and the ACC traces.

Examples

The following example shows sample client traces with DEBUg.

```
debug none fsc soc time
Active client traces - FSC(1) SOC(1)
get a 'user33.abc111' (replace
11:21:33 CG0204 get: F=1 p=FSA ARTW=0000
11:21:33 CG3356 rcvFile: entered
11:21:33 MR1200 set_filename: entered with pathname 'user33.abc111'
11:21:33 CG2078 mvs_rcvFile: entered
11:21:33 CG1944 newGDGname: entered
11:21:33 MV0874 seq_create_file: entered with dsn=USER33.ABC111
11:21:34 MV1545 seq_create_file: ddname=SYS00009
12345678901234567890123456789012345678901234567890123456789012345678901234567890
11:21:34 MV1605 seq create file: data set has recfm=50, lrecl=256, blksize=6233
11:21:34 CG2470 mvs_rcvFile: FASTIO pending for store_type = N
11:21:34 CG2473 mvs_rcvFile: ... recfm=VB, lrecl=256, blksize=6233
11:21:34 CG2560 mvs_rcvFile: request FASTIO with recfmU override for record form
at VB
11:21:34 MF2540 seq_open_file: recfm is VB
11:21:34 MF2598 seq_open_file: BSAM 0 SYS00009 ( )
11:21:34 MF2600 seq_open_file: ncp=29 DA=0 K0=0 DF=0
11:21:34 MF2658 seq_open_file: stream 46EE8 has maxreclen 256
11:21:34 SC0531 initDsConnection: entered
>>> PORT 9,67,113,57,4,32
200 Port request OK.
>>> RETR a
125 Sending data set /u/user33/a
11:21:34 SC0783 accDsConnection: entered
11:21:34 TI3053 WrtStreamFastIO: 0=2 HGPES=00001 BCTE=1000 RLB=50/256/6233
11:21:34 MF2441 seq_close_file: file closed
11:21:34 GV0150 releaseFile: release ddname - SYS00009
11:21:34 GV0171 releaseFile: dynfree() results- rc=0, errcode=0, infocode=0
11:21:34 SC1128 dataClose: entered
250 Transfer completed successfully.
820 bytes transferred in 0.005 seconds. Transfer rate 164.00 Kbytes/sec.
```

```
11:21:34 CU1821 write_smf_record: entered with type 16.  
11:21:34 CU1275 write_smf_record_119: entered with type 16.  
11:21:34 CU2084 write_smf_record: length of smfrecord: 224
```

Usage

By default, DEBUg is off. When the FTP environment is entered, you can activate DEBUg by any of the following methods:

- Use the TRACE or -d parameter on the FTP command.
- Code one or more DEBUg statements in the client's FTP.DATA.
- Code the TRACE statement in the client's FTP.DATA.

Once FTP is started, you can change the DEBUg settings with the DEBUg subcommand.

- The trace supports the DEBUg parameters 1 and 2 that are used with previous product releases. Specifying DEBUg BAS provides the same tracing as parameter 1 formerly provided. The DUMP subcommand now provides the extended tracing that parameter 2 formerly provided.
- The state of the traces points is displayed as a response to the DEBUg subcommand. To see the states without making a change, enter DEBUg ?.
- The setting of the traces is additive as shown by the following:

```
DEBUg NONE CMD  
EZA2851I Active traces: CMD  
DEBUg PAR  
EZA2851I Active traces: CMD PAR
```

- Entering DEBUg with no parameters will toggle the trace on and off. The state of the traces when the trace is toggled off is remembered so that toggling it on restores the previous trace settings. If no traces were active previously, then toggling activates the BAS trace points

```
DEBUg NONE CMD  
EZA2851I Active traces: CMD  
DEB  
EZA2851I Active traces: NONE  
DEB  
EZA2851I Active traces: CMD  
DEB FSC  
EZA2851I Active traces: CMD FSC(1)  
DEB  
EZA2851I Active traces: NONE  
DEB  
EZA2851I Active traces: CMD FSC(1)  
DEB NONE  
EZA2851I Active traces: NONE  
DEB  
EZA2851I Active traces: CMD INT FSC(1) SOC(1)
```

- The timestamp option is demonstrated with the following example:

```
deb fsc(1)  
PC0304 parseCmd: subcommand: deb  
PC0307 parseCmd: parameter 1: fsc(1)  
Active traces: CMD FSC(1)  
Command:  
deb time  
PC0304 parseCmd: subcommand: deb  
PC0307 parseCmd: parameter 1: time  
Active traces: CMD FSC(1)  
Command:  
deb soc(1)  
11:39:37 PC0304 parseCmd: subcommand: deb  
11:39:37 PC0307 parseCmd: parameter 1: soc(1)  
Active traces: CMD FSC(1) SOC(1)  
Command:
```

- For the FSC and SOC trace options only one level of tracing can be defined at any time. However, when level 2 is defined, levels 1 and 2 are active. When level 3 is defined, levels 1, 2, and 3 are active. This progression also applies to levels 4 and 5.

```
deb fsc(2) soc(1)
Active traces: FSC(2) SOC(1)
Command:
deb fsc(1) soc(2)
Active traces: FSC(1) SOC(2)
Command:
```

Tip: The DEBUg FSC command accepts level values 6–8, but provides only level 5 trace data. Likewise, DEBUg SOC accepts level values 4–8, but provides only level 3 trace data.

See [Diagnosing FTP client problems with tracing in z/OS Communications Server: IP Diagnosis Guide](#) for more information about FTP client tracing.

DELEte subcommand—Delete files

Purpose

Use the DELEte subcommand to delete a file on the remote host.

Format

➤ DELEte — *foreign_file* ➤

Parameters

foreign_file

Specifies the name of the file to be deleted on the remote host.

Context

See [Appendix A, “Specifying data sets and files,” on page 359](#) for information about file naming conventions.

DELImit subcommand—Display the file name delimiter

Purpose

Use the DELImit subcommand to display the character that is used as the delimiter between the file name and the file type.

Format

➤ DELImit ➤

Parameters

There are no parameters for this subcommand.

Usage

- The DELImit subcommand should be used for information purposes only.
- You cannot change which character is used as the delimiter.

DIr subcommand—Obtain a list of directory entries

Purpose

Use the DIr subcommand to obtain a list of directory entries or a list of files in a file group on the remote host, or a list of the members of the partitioned data set, as well as auxiliary information about the files.

Format

DIr name (DISK)

Parameters

name

Specifies the name of the directory or file group. The default is the current directory or file group.

(DISK)

Stores the results of the DIr subcommand as data set FTP.DIROUTP in the local current working directory.

- If the local current working directory is an MVS PDS, the member DIROUTP is stored.
- If the local current working directory is a z/OS UNIX directory, the results are stored in a file named diroutp.

Examples

- List the data sets with a common high-level qualifier as the current working directory:

```
EZA1460I Command:
dir
EZA1701I >>> PORT 9,42,105,36,4,70
200 Port request OK.
EZA1701I >>> LIST
125 List started OK
EZA2284I Volume Referred      Ext      Used Recfm Lrecl BlkSz Dsoig Dsname
EZA2284I CPDLB4 2008/10/31      1         1  VB    256  6233  PS  FIFO.DEMO
EZA2284I CPDLB1 2008/11/11      1         2  FB     80  3120  PO  ISPF.ISPPROF
EZA2284I CPDLB3 2000/08/23      1         1  VB    255  3120  PS  LOG.MISC
EZA2284I CPDLB1 1997/01/20      2         2  FB     80  3120  PO  SPF.ISPPROF
EZA2284I CPDLB3 2008/11/11      9         9  VBS   4000  2000  PS  TEST.ABC
EZA2284I CPDLB3 2008/11/11      1         1  VBS   4000  2000  PS  TEST.ABC1
250 List completed successfully.
EZA1460I Command:
```

Note: The DIr output for a RECFM=U data set for the FTP display always shows the same value for lrecl as it shows for blksize.

- List the files for a z/OS UNIX file system directory:

```

cd '/u/user121/ftp.example'

>>>CWD '/u/user121/ftp.example'
250 HFS directory /u/user121/ftp.example is the current working directory
Command:
dir
>>>PORT 9,67,112,25,4,61
200 Port request OK.
>>>NLST
125 List started OK
total 64
-rw-r----- 1 USER121 SYS1      6720 Feb  7 18:48 append02
-rw-r----- 1 USER121 SYS1      3360 Feb  6 18:51 file1
-rw-r----- 1 USER121 SYS1      3883 Feb  6 18:51 file2
-rw-r----- 1 USER121 SYS1      3883 Feb  6 18:51 file3
-rw-r----- 1 USER121 SYS1      7277 Feb  6 18:51 file4
-rw-r----- 1 USER121 SYS1      3360 Feb  6 18:51 file5
250 List completed successfully.
Command:

```

- List the members of a partitioned data set containing load modules:

```

cd 'sys1.linklib'
>>> CWD 'sys1.linklib'
250-Local directory might be a load library
250 "SYS1.LINKLIB" partitioned data set is working directory
Command:
dir d*
>>> PASV
227 Entering Passive Mode (127,0,0,1,4,112)
>>> LIST d*
125 List started OK
Name          Size      TTR      Alias-of AC ----- Attributes ----- Amode Rmode
DD             03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
DELDSD         03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
DELGROU        03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
DELUSER        03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
DG             03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
DMOCIO001      000710 03370C          00 FO          RN RU             31    ANY
DMOCTCTL       000178 033715          01 FO          RN RU             31    ANY
DMOCTFIL       000028 03371D          01 FO          RN RU             31    ANY
DMOCTFMT       00ABC8 033725          01 FO          RN RU             31    ANY
DMOCTL0C       0006E8 03380C          01 FO          RN RU             31    ANY
DMOCTRCE       0008F8 033814          01 FO          RN RU             31    ANY
DMOCTSTR       000588 03381D          01 FO          RN RU             31    ANY
DMODA002       001318 033826          01 FO          RN RU             31    ANY
DMODA003       004618 033909          01 FO          RN RU             31    ANY
DMODA004       000658 033916          01 FO          RN RU             31    ANY
DMODIAG        001E58 03391F          00 FO          RN RU             31    24
DMOVS001       002110 033A04          01 FO          RN RU             31    ANY
DMOVS002       0003D8 033A0F          01 FO          RN RU             31    ANY
DU             03DBD8 031506 IRRENV00 01 FO          RN RU             31    24
250 List completed successfully.
Command:

```

- List the members of a partitioned data set from a text library:

```

cd 'tcpv3.tcpip.profiles'
>>>CWD 'tcpv3.tcpip.profiles'
257 "'TCPV3.TCPIP.PROFILES'" partitioned data set is working directory.
Command: dir
>>>PORT 9,67,112,25,4,32
200 Port request OK.
>>>LIST
125 List started OK.
Name      VV.MM      Created      Changed      Size  Init  Mod  Id
TST6MV1   01.05 1997/06/26 1996/07/10 06:38   16   16   0 USER34
TST6MV2   01.08 1997/05/23 1996/07/03 12:49   16   17   0 USER34
TST6MV3   01.19 1997/05/23 1996/07/10 06:34   16   17   0 USER34
TST6021   01.04 1997/03/04 1996/07/08 09:17   15   15   0 USER34
TST6121   01.10 1997/05/23 1996/07/10 06:26   16   17   0 USER34
250 List completed successfully.
***

```

Usage

- To make a file group the current working directory, use the CD command. The method you use to specify a directory or file group is host-dependent.
- The DIR subcommand provides a complete list of directory entries and gives additional information about the files.

When using this subcommand to list MVS data sets that have a common high-level qualifier as the current working directory on a remote host, the volume names are displayed. However, when displaying a multivolume data set used in an SMS environment, only the first volume name is displayed. To list all volume names for a multivolume data set, issue the following TSO command on the remote host:

```
LISTC ENT('dataset_name') ALL
```

- You can use special characters for pattern matching when specifying the *name*. These characters depend on the host FTP server.
- Special characters you can use for the z/OS FTP server:

A single asterisk by itself indicates that either a qualifier or one or more characters within a qualifier can occupy that position. An asterisk can precede or follow a set of characters. An asterisk as the last qualifier will indicate that 0 or more qualifiers can occupy that position.

A double asterisk indicates that 0 or more qualifiers can occupy that position. A double asterisk cannot precede or follow any characters. It must be preceded and followed by either a period or a blank.

%

A single percent sign by itself indicates that exactly one alphanumeric or national character can occupy that position.

%...%

One to eight percent signs (%) can be specified in each qualifier.

In DIRECTORYMode, ** is not allowed and % and * can be specified in the last qualifier only.

The following shows examples of how the z/OS FTP server special characters can be used.

| Entry | Returns |
|-----------------|---|
| VSAM.DATA.SET | VSAM.DATA.SET <i>only</i> |
| VSAM.DATA.SET% | VSAM.DATA.SET1 VSAM.DATA.SET2 - but not VSAM.DATA.SET30 |
| VSAM.DATA.SET%% | VSAM.DATA.SET30 VSAM.DATA.SET31 - but not VSAM.DATA.SET1 or VSAM.DATA.SET2 |
| VSAM.*.SET | VSAM.DATA1.SET VSAM.DATA2.SET - but not VSAM.DATA.SET.KSDS |

| Entry | Returns |
|---------------|---|
| VSAM.*A | VSAM.A VSAM.BA VSAM.BBA - but not VSAM.B or VSAM.AB |
| VSAM.DATA.* | VSAM.DATA.SET1 VSAM.DATA.SET2 VSAM.DATA.SET.KSDS - but not VSAM.DATA1.SET |
| VSAM.DATA* | VSAM.DATA1 VSAM.DATA23 - but not VSAM.DATA.SET.KSDS |
| VSAM.DATA*. * | VSAM.DATA1 VSAM.DATA23 VSAM.DATA.SET1 VSAM.DATA1.SET VSAM.DATA.SET.KSDS |
| VSAM.** | VSAM VSAM.DATA.SET1 VSAM.DATA.SET2 VSAM.DATA.SET.KSDS - but not VSAM1.DATA.SET |
| **.*DATA | VSAM.DATA NONVSAM.WORK.DATA DATA - but not VSAM.DATA.SET |
| ** | Will return all data sets within the current working directory. If the current working directory is null, this command has the potential to read all available catalogs to which the user has access. This can take a considerable length of time. |

Context

- See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about pattern matching and about specifying data sets and files.
- To get a list containing only the file names in a directory, use the LS subcommand (see [“LS subcommand—Obtain a list of file names ” on page 208](#)).
- To make a file group the current working directory, see [“CD subcommand—Change the directory on the remote host ” on page 149](#).
- To change the local directory, see [“LCd subcommand—Change the local working directory ” on page 175](#).

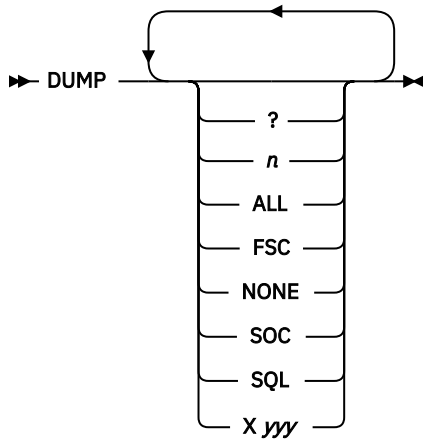
DUMP subcommand—Set extended trace options

Purpose

Use the DUMP subcommand to enable or disable extended internal tracing.

Note: Extended tracing has the potential to generate a large amount of trace data and should not be set unless requested to debug a specific problem in the code.

Format



Parameters

One or more of the following options can be specified:

?

Displays the status of the traces.

n

Specifies the ID number of a specific extended trace point that is to be activated in the FTP code. The number has a range of 1–99.

ALL

This parameter is used to set all of the trace points.

FSC

Activates all of the extended trace points in the file services code. The ID numbers for FSC are 20 to 49.

NONE

This parameter is used to turn off all of the traces.

SOC

Activates all of the extended trace points in the network services code. The ID numbers for SOC are 50 to 59.

SQL

Activates all of the extended trace points in the SQL services code. The ID numbers for SQL are 70 to 79.

Xyyy

This syntax is used to turn off (reset) a trace that is named by yyy. For example, DUMP X21 X22 XSQL will reset the extended trace points 21 and 22 and all of the SQL trace points.

Examples

The following is an example of a dump trace.

```

dump 21 22
Active client dumpIDs - 21 22
get a 'user33.abc111' (replace
12:38:31 MV0456 (21) TU_DSN ....:
0A31D1E4 00020001 000DE4E2 C5D9F3F3 4BC1C2C3 *.USER33.ABC*
0A31D1F4 F1F1F100 00000000 00000000 00000000 *111.....*
0A31D204 - 0A31D223 All zeros (0x20 bytes)12:38:31 MV0512 (21) TU_STATS ....:
0A31D150 00040001 00010100 *.
12:38:31 MV0521 (21) TU_DISP ....:
0A31D158 00050001 00010800 *.
12:38:31 MV2113 (22) RN PA=
0A3296C0 14070000 00000000 0A3296D8 00000000 *.oQ....*
0A3296D0 00000000 *.
12:38:31 MV2115 (22) RN DD=
0A3296B0 00010001 0008E2E8 E2F0F0F0 F0F94B4B *.SYS00009..*
0A3296C0 14070000 0000 *.
12:38:31 MV2117 (22) RN DSN=
0A3296EC 00050001 000DE4E2 C5D9F3F3 4BC1C2C3 *.USER33.ABC*
0A3296FC F1F1F14B 60614B4B 4B4B4B4B 4B4B4F6B *111.-/.....,*
0A32970C 6C6D6E6F 4B4B4B4B 4B4B4B4B 4B4B7A7B *%>?.....:##
0A32971C 7C7D7E4B *@'=.*
>>> PORT 9,67,113,57,4,59
200 Port request OK.
>>> RETR a
125 Sending data set /u/user33/a
250 Transfer completed successfully.
820 bytes transferred in 0.005 seconds. Transfer rate 164.00 Kbytes/sec

```

Usage

The setting of the traces is additive. This is demonstrated by the following example:

```

dump none 21
EZA2850I Active dumpIDs: 21
dump 22
EZA2850I Active dumpIDs: 21 22

```

Entering dump with no parameters is the same as entering dump with the ? parameter.

The range of 99 extended trace points is defined to allow easy extension of the trace points by the TCP/IP service team. Additional trace points can be added to the code without any changes to the external mechanism to control the traces.

See [Diagnosing FTP client problems with tracing in z/OS Communications Server: IP Diagnosis Guide](#) for more information about FTP client tracing.

EBcdic subcommand—Change the data transfer type to EBCDIC

Purpose

The EBcdic subcommand enables you to change the data transfer type to EBCDIC.

Format

➤ EBcdic ➤

Parameters

There are no parameters for this subcommand.

Usage

Use the EBcdic subcommand to direct FTP to transfer data with no translation.

Context

For more information about transfer methods, see [Table 13 on page 46](#).

EUckanji subcommand—Change the data transfer type to EUCKANJI

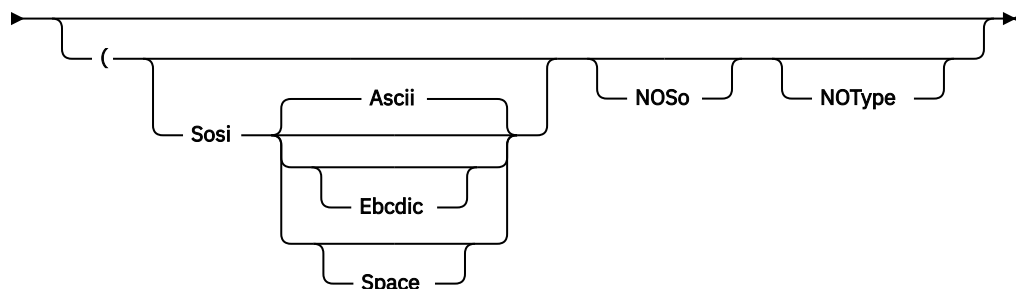
Purpose

Use the EUckanji subcommand to change the data transfer type to Extended UNIX Code (EUC) kanji.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the Site/LOCSite SBDataconn or Site XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format

➤➤ EUckanji ➡



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters – Ascii, Ebcdic or Space. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to and from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding Type command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The EUckanji client subcommand is equivalent to the TYPE B 2 server command.

Context

See [“FTP with traditional DBCS support” on page 78](#) and [“Support for MBCS languages” on page 81](#) for more information.

FEature subcommand—Query FTP server for features it supports

Purpose

Ask the FTP server which features it supports. FTP clients use this command to determine which languages the server supports, and which features the server supports.

Format

► FEature ◄

Parameters

There are no parameters for this subcommand.

Usage

The minimum abbreviation for the feature subcommand is fe.

File subcommand—Set the file structure to File

Purpose

Use the File subcommand to set the file structure to File. This is equivalent to specifying the STRucture F subcommand. See [“STRucture subcommand—Set the file structure” on page 286](#) for more information.

Format

► File ◄

Parameters

There are no parameters for this subcommand.

Get subcommand—Copy files

Purpose

Use the Get subcommand to copy a file from the remote host to your local host.

Format

► Get — *foreign_file* — *local_file* (— REPLACE —) ◄

Parameters

foreign_file

Specifies the name of the file to be retrieved from the remote host.

local_file

Specifies the name of the local file created as a result of the Get subcommand.

If the current local working directory is a PDS, *local_file* is the name of the member in the PDS. If the current local working directory is a data set prefix, the local file is a sequential data set with the *local_file* name appended to the current local working directory. If the current local working directory is a z/OS UNIX file system directory, the local file is a z/OS UNIX file in that directory.

You can override the use of the current local working directory in the local file name by specifying the *local_file* value as a complete data set name enclosed in single quotation marks ('). If *local_file* is not specified, the *local_file* name is the same as the *foreign_file* name.

The following apply when the *local_file* value specifies a new file in a z/OS UNIX directory:

- The UNIXFILETYPE configuration option specifies whether the FTP client creates a regular file or a named pipe.
- The UMASK configuration option specifies the file permissions of the new file or named pipe.

Rule: When the *local_file* value specifies an existing named pipe in a z/OS UNIX directory, you must configure UNIXFILETYPE FIFO before you start the file transfer.

(REPLACE

Causes the *local_file* value on your local host to be overwritten if the value is an existing MVS data set or z/OS UNIX regular file. If the MVS data set or z/OS UNIX regular file already exists, and you do not use the (REPLACE parameter, the existing data set is not overwritten. A message informing you of this is displayed.

If the *local_file* value is an existing MVS data set and you specify the (REPLACE option, the data in the file is overwritten, but not reallocated; the local data set retains its existing characteristics.

If the *local_file* value is an existing z/OS UNIX named pipe, the (REPLACE option is not allowed.

Results:

- FTP uses either the characteristics of the local file, if it exists, or uses the values specified with the LOCStite subcommand. Characteristics of the transmitted (foreign file) data set are unknown.

When you use the Get subcommand, FTP might truncate data records and you might lose data, if one of the following occurs:

- If you are creating a new data set at the client and the value of LRecl, as shown by the LOCStat command, is a value less than the LRecl of a received data set, then FTP truncates the received data set.
- If the data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, then FTP truncates the transmitted data set.

You could also encounter truncated data records or lost data when you use the Get subcommand with the REPLACE option.

A Get subcommand that the system issues for the following foreign files erases the contents of the existing local data set:

- An empty foreign file
- A foreign file that does not exist
- A foreign file that another process holds
- If FTP does not support directory content transfers in partitioned data sets, it is not possible to FTP load modules.
- If the data set is migrated, it is replaced regardless of the replace option.

- When the local file is a named pipe on your local host, the following apply:
 - FTP cannot open the file until you start a process to read from the named pipe. If FTP is the first process to open the named pipe, it blocks until another process opens the named pipe for reading, or until the FIFOOPEN TIME timer expires.
 - The remote file is appended to the local file.
 - If the name specified for *local_file* is not acceptable to your local host, the file is not transferred.
 - To get a file from the remote host, you must have a defined working directory on that host and you must have read privileges to the files in this working directory.
 - If the data set has been preallocated, you must specify DSORG=PS on the DCB statement in the JCL.
 - When a PDS or PDSE member is transmitted, the user data associated with the PDS member is also transferred to the directory on the target host if the following conditions are true;
 - Data is in block or compressed data transfer mode
 - Data has a representation type of EBCDIC
 - Transfer is from one MVS directory to another
- No PDS directory information is transferred if the member is null (empty).

Related topics:

- See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about naming conventions.
- See [“CD subcommand—Change the directory on the remote host ” on page 149](#) and [“ACCT subcommand—Supply account information ” on page 144](#) for more information about working directories.
- Get can be used with the PROXY subcommand to transfer files from a host on a primary connection to a host on a secondary connection. See [“PROXY subcommand—Execute FTP subcommand on secondary control connections” on page 235](#) for more information.
- See [“Using z/OS UNIX System Services named pipes” on page 104](#) for more information about storing data into named pipes.
- See [“LOCSITE subcommand—Specify site information to the local host” on page 179](#) and the UNIXFILETYPE (FTP client and server) statement information in [z/OS Communications Server: IP Configuration Reference](#) for more details about the UNIXFILETYPE configuration option.

GLob subcommand—Toggle expansion of metacharacters

Purpose

Use the GLob subcommand to toggle globbing (the expansion of metacharacters in file names) for the MDelete, MGet, and MPut subcommands.

Format

➤ GLob ➤

Parameters

There are no parameters for this subcommand.

Examples

Assume that the files m1 and m1* exist in the directory /u/user33/mpp1.

```
Command:
pwd
>>> PWD
257 "/u/user33/mpp1" is the HFS working directory

Command:
lpwd
Local directory name set to hierarchical file /u/user33

Command:
prompt
Interactive mode is off
```

```
Command:
mget m1*
>>> PORT 9,67,113,57,4,43
200 Port request OK.
>>> NLST m1*
125 List started OK
250 List completed successfully.
>>> PORT 9,67,113,57,4,44
200 Port request OK.
>>> RETR m1
125 Sending data set /u/user33/mpp1/m1
250 Transfer completed successfully.
200 bytes transferred in 0.050 seconds. Transfer rate 4.00 ...
Kbytes/sec.
>>> PORT 9,67,113,57,4,45
200 Port request OK.
>>> RETR m1*
125 Sending data set /u/user33/mpp1/m1*
250 Transfer completed successfully.
200 bytes transferred in 0.020 seconds. Transfer rate 10.00 ...
Kbytes/sec.
```

```
Command:
delete /u/user33/m1
>>> DELE /u/user33/m1
250 /u/user33/m1 deleted.
```

```
Command:
delete /u/user33/m1*
>>> DELE /u/user33/m1*
250 /u/user33/m1* deleted.
```

```
Command:
glob
Globbing off
```

```
Command:
mget m1*
>>> PORT 9,67,113,57,4,46
200 Port request OK.
>>> RETR m1*
125 Sending data set /u/user33/mpp1/m1*
250 Transfer completed successfully.
200 bytes transferred in 0.010 seconds. Transfer rate 20.00 ...
Kbytes/sec.
```

With globbing off, at most one file will match the pattern. Also, the NLST command is not sent to look for pattern matches.

Usage

GLob acts as a toggle that turns metacharacter expansion on or off. By default, GLob is on.

Context

For more information about globbing, see the [z/OS UNIX System Services User's Guide](#).

HAngeul subcommand—Change the data transfer type to HANGEUL

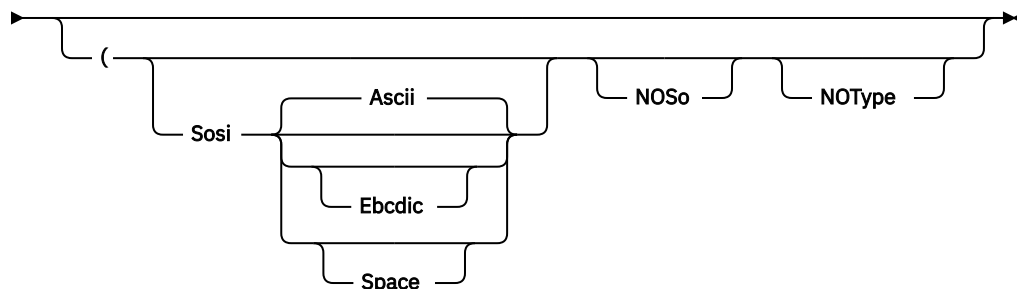
Purpose

Use the HAngeul subcommand to change the data transfer type to Hangeul.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SIte/LOCSite SBDataconn or SIte XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format

➡ HAngeul ➡



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters – Ascii, Ebcdic or Space. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The HAngeul client subcommand is equivalent to the TYPE B 5 server command.

Context

See [“FTP with traditional DBCS support”](#) on page 78 for more information.

Parameters

(NOType

Suppresses sending of the TType command for host servers that do not support this data transfer type.

Usage

This subcommand causes no conversion to be performed on the transferred file. It has exactly the same effect as the EBcdic TType command alias.

Context

See “FTP with traditional DBCS support” on page 78 for more information.

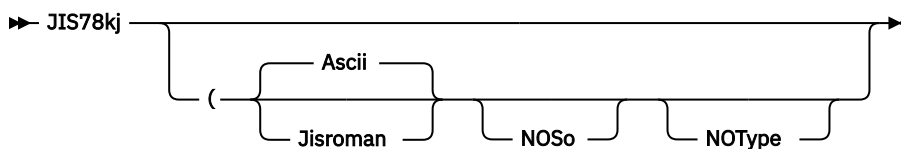
JIS78kj subcommand—Change the data transfer type to JIS78KJ

Purpose

Use the JIS78kj subcommand to change the data transfer type to JIS78KJ (1978 edition).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SItE/LOCStE SBDatConn or SItE XLatE subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Ascii

Use ASCII shift-in escape sequence ESC (B in the transferred data.

If neither Ascii nor Jisroman is specified, the ASCII shift-in sequence is used.

Jisroman

Use Jisroman shift-in escape sequence ESC (J in the transferred data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

- The JIS78kj or JIS78kj (ASCII client subcommands are equivalent to the TYPE B 4 A server command.
- The JIS78kj (JISROMAN client subcommand is equivalent to the TYPE B 4 R server command.
- The JIS78kj (JISROMAN NOSO client subcommand is equivalent to the TYPE B 4 R N server command.

Context

See “FTP with traditional DBCS support” on page 78 and “Support for MBCS languages” on page 81 for more information.

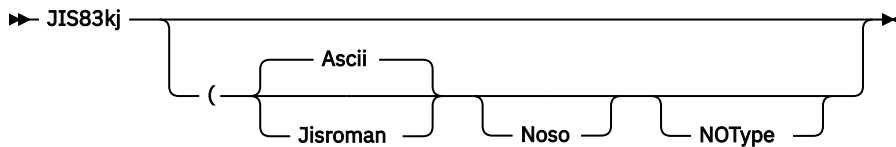
JIS83kj subcommand—Change the data transfer type to JIS83KJ

Purpose

Use the JIS83kj subcommand to change the data transfer type to JIS83KJ (1983 edition).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SItE/LOCStE SBDataconn or StE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Ascii

Use ASCII shift-in escape sequence ESC (B in the transferred data.

If neither Ascii nor Jisroman is specified, the ASCII shift-in sequence is used.

Jisroman

Use Jisroman shift-in escape sequence ESC (J in the transferred data.

NOSO

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding Type command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

- The JIS83kj or JIS83kj (ASCII client subcommands are equivalent to the TYPE B 3 A server command.
- The JIS78kj (JISROMAN client subcommand is equivalent to the TYPE B 3 R server command.

Context

See [“FTP with traditional DBCS support” on page 78](#) and [“Support for MBCS languages” on page 81](#) for more information.

Ksc5601 subcommand—Change the data transfer type to KSC-5601

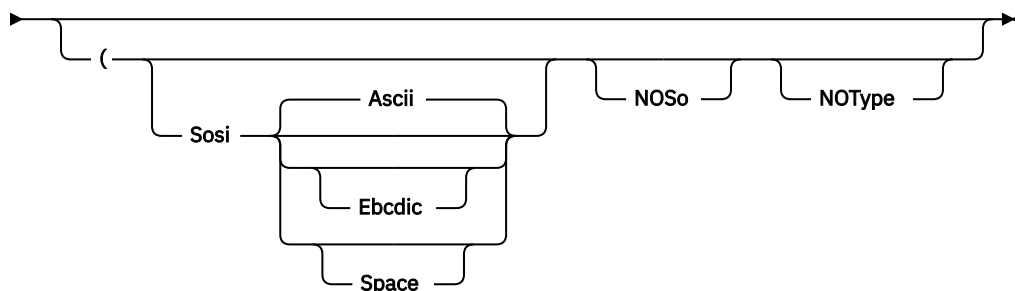
Purpose

Use the Ksc5601 subcommand to change the data transfer type to KSC-5601.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SItE/LOCStE SBDataconn or StE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format

►► Ksc5601 ►►



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — Ascii, Ebcdic or Space. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The Ksc5601 client subcommand is equivalent to the TYPE B 6 server command. See [“FTP with traditional DBCS support”](#) on page 78 for more information.

Context

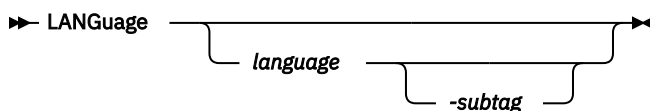
See [“Support for MBCS languages”](#) on page 81 for more information.

LANGuage subcommand—Set the language used for FTP replies from the server

Purpose

Ask the server to use a certain language for FTP server replies, or reset the language in use to the server's default language.

Format



Parameters

language

A 2-character code as specified by RFC 1766 identifying the language to be used for FTP replies. RFC 1766 defines these codes as those listed in ISO 639. See [Appendix D, “Related protocol specifications,”](#) on page 381 for information about accessing RFCs.

-subtag

A 2-character code as specified by RFC 1766 identifying a variation of language. RFC 1766 defines these codes as ISO 3166 alpha-2 country codes. See [Appendix D, “Related protocol specifications,”](#) on page 381 for information about accessing RFCs.

Results

- The LANGUage *language -subtag* subcommand is equivalent to the LANG *language -subtag* command.
- LANGUage with no parameters sets the language for FTP replies to the server's default language. For most FTP servers, the default is US English encoded in 7-bit ASCII.
- LANGUage with a parameter requests the server to use *language* for FTP replies. As specified in RFC 2640, it also directs the server and client to use UTF-8 encoding of path names on the control connection.

Rules:

- This subcommand is not available unless EXTENSIONS UTF8 is encoded in FTP.DATA.
- The subcommand is not available if you have disabled UTF-8 encoding with a LOCSite command or FTP start option.

Guideline: Use the FEature subcommand to determine which languages the server offers prior to using the LANGUage subcommand to request a language for FTP replies. See [“FEature subcommand—Query FTP server for features it supports”](#) on page 166 for information about using the FEature subcommand.

LCd subcommand—Change the local working directory

Purpose

Use the LCd subcommand to change the current working directory on the local host.

Format

- In a TSO Environment:

```
LCd — qualifier —>
```

- In a z/OS UNIX System Services Environment:

```
LCd — qualifier —>
```

Parameters

qualifier

Specifies either a common prefix for a group of sequential data sets or the qualifiers of a PDS.

Note: In a z/OS UNIX environment, you can omit the qualifier on the LCD subcommand. Doing so changes the current working directory to your home directory. If you do not have a home directory, the working directory is not changed, and no message is issued.

Examples

- Change the local current working directory:

```
lcd ftp.test1
```

```
Local directory name set to partitioned data set USER14.FTP.TEST1.  
Command:
```

- When the LCD subcommand is issued, *qualifier* is appended to the current local working directory. For example, if the current local working directory is TCPUSR14 . TEST and you issue the LCD subcommand LCD FILES, the new working directory becomes TCPUSR14 . TEST . FILES.
- To override the existing directory rather than append to the directory, issue the *qualifier* in single quotation marks ('). For example, if the current local working directory is TCPUSR14 . TEST and you issued the LCD subcommand LCD 'FTP . FILES', the new working directory is FTP . FILES.
- If a PDS exists with the exact name of the current local working directory, FTP considers the working directory to be that PDS. Otherwise, FTP considers the working directory to be a common prefix qualifier for sequential data sets.

If a PDS exists with the same name as the current local working directory, but you want the current local working directory to be treated as a common prefix for sequential data sets, specify the working directory with a period (.) at the end. For example, if a PDS named TCPUSR14 . TEST exists, the subcommand LCD 'TCPUSR14 . TEST' makes the PDS TCPUSR14 . TEST the current local working directory. A subsequent Get command used to copy data set name1 would add the member name1 to the TCPUSR14 . TEST PDS. In contrast, the statement LCD 'TCPUSR14 . TEST .' would make the current local working directory TCPUSR14 . TEST ., a prefix for sequential data sets. A subsequent Get command used to copy data set name1 would create the sequential data set TCPUSR14 . TEST . name1.

- To back up one level of the current local working directory, issue the LCD subcommand with two periods (..) at the end. For example, if the working directory is jones . source, the subcommand LCD .. makes jones . the working directory.

Usage

When you enter an FTP session, the working directory on the local host is set according to the environment in which the FTP client is invoked: \$HOME in z/OS UNIX, your MVS user ID in TSO.

Testing throughput with *DEV.NULL

You can use the Get or MVSGet subcommand to copy many files (or one large file) without storing the files in the client's file system. This is useful for testing purposes because you do not have to worry about allocating the disk space on the client system.

Procedure

1. Change the working directory to *DEV.NULL by entering the following subcommand:

```
LCD *DEV.NULL
```

This affects the working directory for only the Get or MVSGet subcommand.

2. Use the Get or MVSGet subcommand to copy the file to the client system. The input data set must be valid, and the output file can be a new or an existing file. In either case, the file is not actually stored. The following response shows information such as the number of bytes transferred and the rate of transfer.

```

Command:
lcd *dev.null
Working Directory for GET is NULL Device
for PUT is HFS directory /tmp
Command:
get 'user2.junk(junk)' example
>>> EPSV
229 Entering Extended Passive Mode (|||1034|)
>>> RETR 'user2.junk(junk)'
125 Sending data set USER2.JUNK(JUNK)
250 Transfer completed successfully.
65 bytes transferred in 0.070 seconds. Transfer rate 0.93 Kbytes/sec.
Command:

```

3. To end the use of the *dev.null directory for the Get or MVGet subcommand, issue another change local working directory subcommand.

LMkdir subcommand—Create a directory on the local host

Purpose

Use the LMkdir subcommand to create a PDS, PDSE, or z/OS UNIX file system directory on the local host. This subcommand provides you with an easy way to create a directory in the local host for data transfer.

Format

```

➤ LMkdir — directory ————— ➤
      ( — like — remote_directory — )

```

Parameters

directory

Specifies the name of the PDS, PDSE, or z/OS UNIX file system directory to be created.

remote_directory

Specifies the name of a remote MVS PDS or PDSE that is to be a model for the directory to be created. This parameter is valid only when *directory* is a PDS or PDSE name. If you specify this parameter, the local site variables will change, and FTP will open and read the remote data set.

Examples

In this example, before LMkdir is issued, the local host had the following data sets:

- MVSUSER.ISPF.ISPPROF
- MVSUSER.JCL.CNTL
- MVSUSER.SMFTEST
- MVSUSER.TCPIP.DATA

```

User:  ftp 1.1.2.3
System: IBM FTP CS/390 V2R10  1997  314 01:11 UTC
        220-EZAFTSRV IBM CS/390 V2R10 at EMU.ABC.OZ, 17:04:12 on 08/03/97
        220 Connection will close if idle for more than 5 minutes.
        NAME (<host>:tsouserid):

User:  mvsuser
System: >>>USER mvsuser
        331 Send password please.
        Password:
        >>>PASS *****
        230 MVSUSER is logged on.
Command:

```

```

User: lpwd
System: Local directory is MVSUSER.
        Command:
User: lcd ftp
System: Local directory name set to MVSUSER.FTP.
        Command:
User: lmkdir example
System: MVSUSER.FTP.EXAMPLE created.
        Command:

```

MVSUSER.FTP.EXAMPLE has now been created. You can the same result directly with the LMKDIR 'MVSUSER.FTP.EXAMPLE' command.

This example illustrates the use of the (like parameter:

```

Command:
lmkdir 'mvsuser.example.linklib' (like 'sys1.linklib'
>>> XDSI 'sys1.linklib'
200 SITE PDSTYPE=PDS RECFM=U BLKSIZE=32760 DIRECTORY=800 LRECL=0 PRIMARY=482 SECONDARY=30
CYLINDERS
local site variables have changed
MVSUSER.EXAMPLE.LINKLIB created.
Command:

```

After the LMkdir subcommands were issued, the local host had the following data sets under MVSUSER:

- MVSUSER.FTP.EXAMPLE
- MVSUSER.ISPF.ISPPROF
- MVSUSER.JCL.CNTL
- MVSUSER.SMFTEST
- MVSUSER.TCPIP.DATA
- MVSUSER.EXAMPLE.LINKLIB

Usage

- FTP provides no subcommand to display a list of local directory entries. You should use TSO ISPF facility to check whether the directory is created by the LMkdir subcommand.
- If you are running FTP in a z/OS UNIX environment, you can use the ! subcommand to check the status of z/OS UNIX file system directories.
- The *directory* value is appended to the local current working directory to form the name of the created PDS, PDSE, or z/OS UNIX file system directory. To override the local current working directory, specify an absolute z/OS UNIX file system path name:

```
/directory
```

or a fully qualified name in quotes:

```
'directory'
```

When *directory* is a PDS or a PDSE name, the data set characteristics of the newly allocated PDS or PDSE are determined by the settings of the local site variables.

- You can use the (like parameter of the lmkdir subcommand to specify a PDS or PDSE on the server host that has characteristics you want the local directory to have.
 - The (like option is valid only when both client and server are z/OS V1R5 or later, and the working directory for both the client and server is an MVS high level qualifier (HLQ).
 - If you use the (like parameter, the local site variables will change.
 - The FTP client will set the local site variables for you so the new PDS or PDSE is created with characteristics similar to the remote directory. The client, however, can approximate only certain characteristics of the data set such as space type, primary, and secondary. For complete control over these characteristics, do not use the (like parameter.

- FTP must open and read the remote directory to determine its characteristics. If this is not acceptable, do not use the (like parameter.
- Only the 3390 device architecture is supported. If the (like parameter is used for directories residing on other types of devices, unpredictable results will occur. Use of the (like parameter when the source or target directories do not reside on a 3390 architecture device is not recommended.
- If the remote data set is migrated, the server will inspect the AUTORECALL setting to determine whether to recall the data set or fail the request. If AUTORECALL is true, FTP will attempt to recall the data set; otherwise it will fail the request. Similarly, if the remote data set is not mounted, the server will inspect the AUTOMOUNT setting to determine whether to mount the data set or fail the request. If AUTOMOUNT is true, the server will attempt to mount the data set; otherwise, it will fail the request. You can change the server's AUTOMOUNT and AUTORECALL settings with the SITE subcommand. Choosing AUTOMOUNT or AUTORECALL could result in a long delay as the server waits for the data set to become available.

Related Topics:

- See “Dynamic allocation of new data sets” on [page 82](#) for information about FTP configuration options that determine data set characteristics when creating a PDS or PDSE directory.
- See “[LOCSite subcommand—Specify site information to the local host](#)” on [page 179](#) and “[LOCStat subcommand—Display local status information](#)” on [page 201](#) for information about setting and displaying FTP configuration options.
- See “[Site subcommand—Send site-specific information to a host](#)” on [page 247](#) for information about setting the AUTOMOUNT and AUTORECALL values.

LOCSite subcommand—Specify site information to the local host

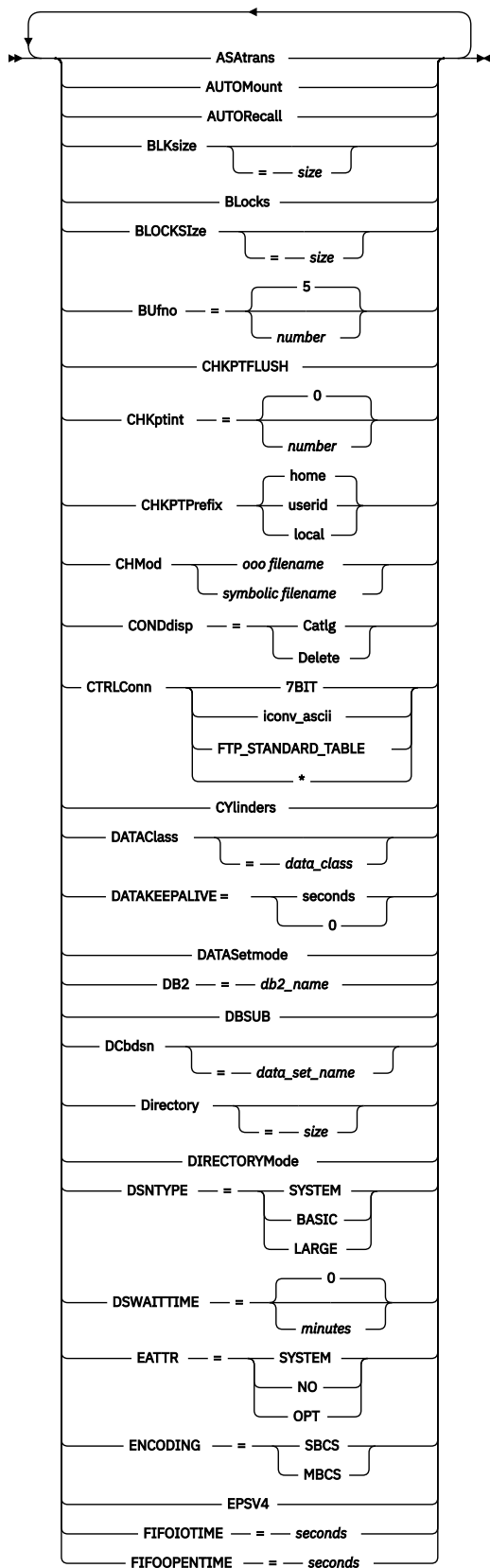
Purpose

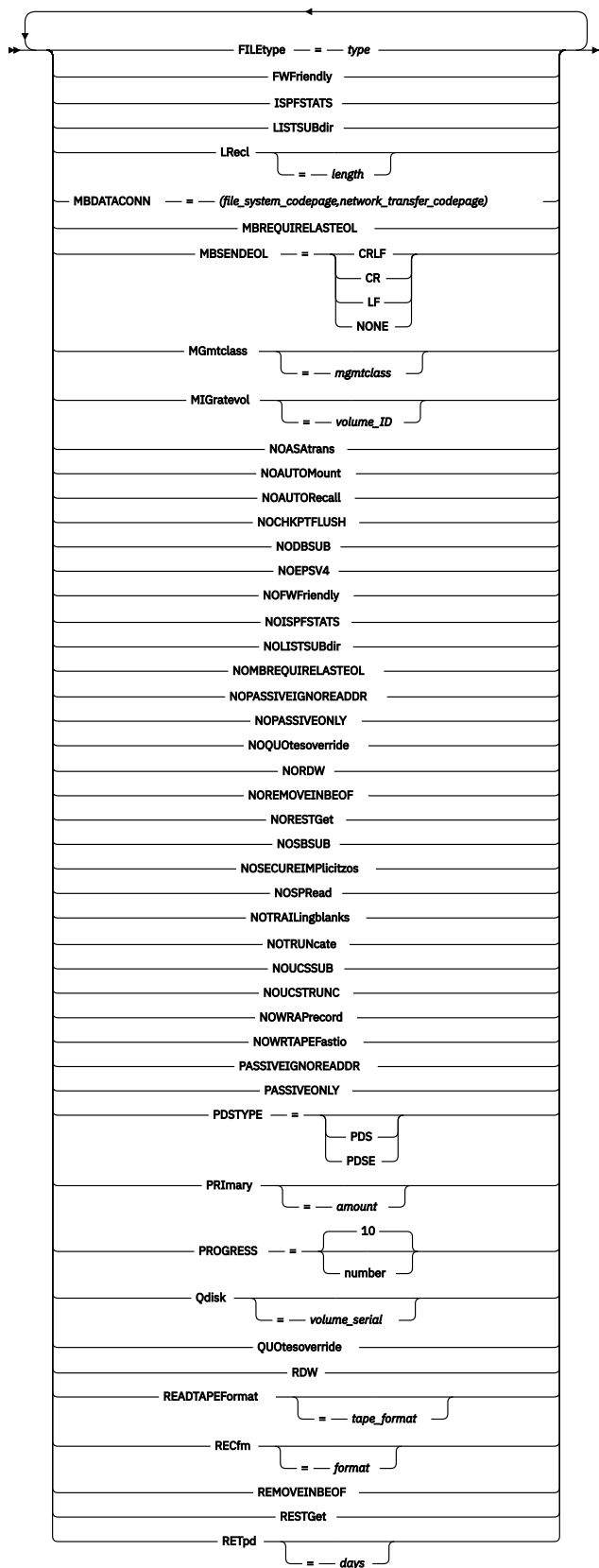
Use the LOCSite subcommand to specify information that is used by the local host to provide services specific to that host system.

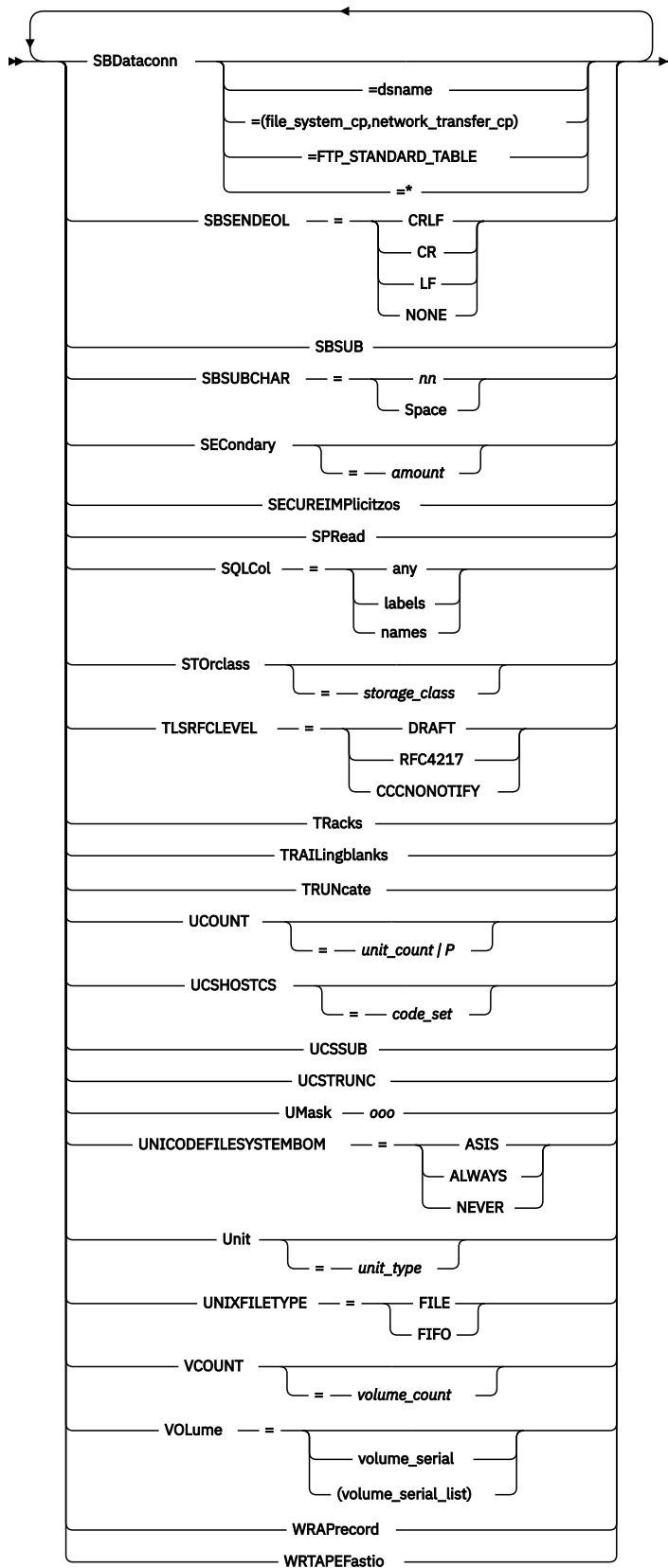
Format

➤ LOCSite — option ➤

options







Parameters

ASAtans

Permits the FTP client to interpret characters in the first column of ASA files being transferred as print control characters.

AUTOMount

Permits automatic mounting of volumes for data sets on volumes that are not mounted. If AUTOMount is specified and an unmounted volume is needed, a message is automatically issued to the MVS operator console requesting that the volume be mounted. The MVS operator must then mount the volume and reply to the message before FTP can proceed.

AUTORecall

Permits automatic recall of migrated data sets.

BLKsize

Specifies the block size of a newly allocated data set. BLKsize is functionally equivalent to BLOCKSize. The BLOCKSize parameter is obsolete, but it is accepted to provide compatibility with previous releases of z/OS TCP/IP.

When specified without a *size*, no block size is used when allocating the new data set. When specified without a *size*, the equal sign (=) is optional.

Specify BLKsize with no value if you are also specifying `DATAclass=data_class` and you want the SMS data class to provide the BLKsize value, or if you are specifying `DCbdsn=data_set_name` and you want to use the block size from the DCBDSN data set. If `BLKsize=size` is specified with either the DATAclass or DCbdsn parameter, the value specified by the LOCSite BLKsize parameter overrides the DATAclass or DCbdsn block size.

Notes:

1. If you specify BLKsize without a *size*, FTP does not specify the block size when allocating new data sets.
2. Be especially careful specifying both BLKsize= and Blocks. While there are conditions where this is tolerated, if a valid BLKsize cannot be determined, the data set will not be created when the allocation is attempted.

size

Specifies the block size of a newly allocated data set. The valid range is 0–32 760.

BLKsize=0 is a special case. When BLKsize=0 is specified, the operating system attempts to determine a block size for the new data set. FTP does not create the new data set unless the system is able to establish a nonzero block size.

BLocks

Specifies that primary and secondary space allocations are in blocks.

If both PRImary and SECondary are specified as 0, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the BLocks parameter is ignored.

BLOCKSize

Specifies the block size of a newly allocated data set. BLOCKSize is functionally equivalent to BLKsize. BLOCKSize is obsolete but it is accepted to provide compatibility with previous releases of z/OS TCP/IP. See the BLKsize parameter for more information.

BUfno

Specifies the number of access method buffers that is used when data is read from or written to a data set. The valid range is 1–35. The default value is 5.

CHKPTFLUSH

Specifies that z/OS flushes each checkpoint record from the buffer to the storage media after this record is written into the buffer.

CHKptint

Specifies the checkpoint interval that the FTP client uses when you issue the APPEnd, Put, and MPut subcommands. When RESTGET is TRUE, the CHKptint parameter also specifies the checkpoint

interval that the FTP client uses when you issue the Get and MGet subcommands. For details about the RESTGET value, see [RESTGet](#) or the [RESTGET statement \(FTP client\)](#) information in [z/OS Communications Server: IP Configuration Reference](#).

The checkpoint interval is the number of records that are sent between restart markers when transferring files in EBCDIC block mode or EBCDIC compress mode. If the checkpoint interval is 0, no checkpointing occurs and no restart markers are transmitted. The default value is 0.

Do not set CHKptint to a value greater than 0 at the FTP client unless the server supports the REStart command and can process checkpoint markers in the file transfer data stream.

Results: A CHKptint value greater than 0 enables checkpointing for file transfers that meet the following conditions:

- Type is EBCDIC
- Mode is Block or Compressed
- Filetype is SEQ
- The subcommand that was entered was APpend, MPut, or Put, and RESTGET is FALSE
- The subcommand that was entered was APpend, MPut, Put, Get, or MGet, and RESTGET is TRUE.

Checkpointing never occurs when the local file is a z/OS UNIX named pipe.

number

Specifies the checkpoint interval for the sending site in a file transfer request. This value is used to determine when checkpoint marker blocks are to be transmitted so that transmission can be restarted based on the information in the last marker.

A large checkpoint interval means that a large amount of data is sent between markers and few markers are sent. A smaller checkpoint interval means that less data is sent in between markers and therefore more markers are sent.

The costs involved with using a nonzero checkpoint interval are:

- The markers themselves are transmitted, which means that more bytes are being sent across the network (approximately 44 bytes per marker).
- For each marker that is sent by the client, a reply must be sent by the server on the control connection. The reply acknowledges the marker and provides the corresponding marker for the server.

To estimate the appropriate checkpoint interval, use the following formula. You need to know the record length of the file you are transferring and how much data you think can be transmitted reliably.

$$\text{CHKPTINT} = \frac{\text{amount of data in interval}}{\text{record length of the file}}$$

Do not use a CHKptint more often than once every 200 KB of data sent. For example, if the file you are transferring has 80-byte records, the checkpoint interval is 2560:

$$\begin{aligned}\text{CHKPTINT} &= 200\text{KB} / 80 \text{ bytes} \\ &= 200 * 1024 \text{ bytes} / 80 \text{ bytes} \\ &= 2560\end{aligned}$$

CHKPTPrefix

Specifies a key which is used to determine the *hlq* of the checkpoint data set. The name will be *hlq.FTP.CHKPOINT* or *hlq.pds_name(CHKPOINT)*.

home

The default.

userid

Creates the data set 'userid.FTP.CHKPOINT'.

local

Creates a file named `current_path.FTP.CHKPOINT` or, if the path happens to be a PDS, then the file name is `current_path.pds_name(CHKPOINT)`.

Note the exceptions for z/OS UNIX and BATCH jobs in the following chart using 'path' for current path and 'FN' for FTP.CHKPOINT:

| Parameter | z/OS UNIX | Batch | TSO |
|----------------------|-----------|---------------------------|---------------------------|
| Home | path.FN | userid.FN | tso_prefix.FN |
| UserId | userid.FN | userid.FN | userid.FN |
| Local | path.FN | path.FN | path.FN |
| Local with path=pdse | N/A | path.pds_na (CHKPOINT) | path.pds_na (CHKPOINT) |

CHMod

Changes the permission bits for a file.

ooo filename

ooo is an octal mask representing the permissions you want to assign to *filename*. Form the octal mask by OR'ing the constants corresponding to the permission bits you want set:

400

User read

200

User write

100

User execute (or list directory)

040

Group read

020

Group write

010

Group execute

004

Other read

002

Other write

001

Other execute

You cannot use the LOCSITE subcommand CHMod parameter to set the following permission bits:

- Set-user-ID bit
- Set-group-ID bit
- Sticky bit

See the [z/OS UNIX System Services User's Guide](#) and the [z/OS UNIX System Services Command Reference](#) for more information about file permissions.

symbolic filename

symbolic represents the permissions you want to apply to *filename*.

Note: *symbolic* is specified as follows:

```
{u|g|o|a}{=|+|-}{r|w|x|rw|rx|wx|wrx}
```

where u, g, o, a, =, +, -, r, w, and x are as defined for the z/OS UNIX chmod command.

If *filename* does not begin with a slash character (/), it is appended to the current working directory. If *filename* does begin with a slash character (/), it is interpreted as a complete directory name.

The file name specified must be a z/OS UNIX file name for a single file and cannot contain a wildcard (*) for multiple files. The setting of QUOTESOVERRIDE is ignored and all quotation marks are treated as part of the file name.

The CHMOD keyword must be the only or last keyword on a LOCStite subcommand.

CONDdisp

Specifies the disposition of the data set if a retrieve operation for a new data set ends before all of the data is written.

Catlg

Specifies that a data set is kept and cataloged when an FTP file transfer ends prematurely.

Delete

Specifies that a data set is deleted when an FTP file transfer ends prematurely.

Delete is ignored if the file transfer failed as a result of the FTP client being terminated or if the client has received checkpoint information during data transfer.

CTRLConn

Specifies the ASCII code page to be used for control connections. The valid subcommands are:

```
LOCStite CTRLConn=7BIT
LOCStite CTRLConn=iconv_ascii
LOCStite CTRLConn=FTP_STANDARD_TABLE
LOCStite CTRLConn=*
```

See [“Support for SBCS languages”](#) on page 77 for more information.

7BIT

Indicates 7-bit ASCII is to be used.

iconv_ascii

Is a name recognized by iconv to indicate an ASCII code page. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

FTP_STANDARD_TABLE

Specifies that the FTP internal tables, which are the same as the tables that are shipped in TCPXLBIN(STANDARD), are to be used on the control connection.

Specifies that the ASCII used at initialization is to be used.

Note: Setting the control connection code page using LOCStite CTRLCONN disables UTF-8 encoding of the control connection. You must start the client again to restore UTF-8 encoding.

CYlinders

Specifies that primary and secondary space allocations are in cylinders.

If both PRImary and SECondary are specified as 0, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the CYlinders parameter is ignored.

DATAclass

Specifies the SMS data class, as defined by your organization, for the target host. Specifying DATAclass with no parameter value cancels the dataclass specification. The equal sign (=) is optional in this case.

See [“Specifying values for new data sets”](#) on page 82 for more information about specifying attributes when allocating new data sets.

data_class

Specifies the SMS data class, as defined by your organization, for the target host. If values are specified for any of the following LOCStite parameters, the values specified by the LOCStite parameter override the value specified in the SMS dataclass:

- BLKsize

- Directory
- LRecl
- PRImary
- RECfm
- RETpd
- SEConDary

If the DCbdsn parameter is specified on the LOCSite subcommand, the LRecl, RECfm, BLOCKSize, and RETpd (if specified) of the DCBDSN data set overrides the values specified in the data class.

If the MGmtclass parameter is specified on the LOCSite subcommand, and the requested management class specifies a retention period, the retention period value of the management class can override the retention period value of the dataclass.

DATAKEEPAIVE

Specifies the data connection keepalive timer value for the FTP client.

seconds

The number of seconds that elapse before a keepalive packet is sent on the FTP data connection. Valid values are in the range 60-86400 or 0. If you specify the value 0, the DATAKEEPAIVE timer is disabled. For passive mode data connections, the keepalive timer that you configured in PROFILE.TCPIP controls how often keepalive packets flow on the data connection. For active mode data connections, FTP suppresses the PROFILE.TCPIP keepalive timer.

Result: Specifying a DATAKEEPAIVE value prevents a network device from closing the data connection during periods of inactivity on the data connection.

DATASetmode

Specifies that all the data set qualifiers located below the current directory are treated as entries in the directory (disables DIRECTORYMode).

DB2

Specifies the name of the Db2 subsystem.

db2_name

The name of the Db2 subsystem.

DBSUB

Specifies that substitution is allowed for data bytes that cannot be translated in a double-byte character translation. The substitution character is selected by the C/C++ iconv() function; see information about Locales and Character Sets in [z/OS XL C/C++ Programming Guide](#) for more details.

DCbdsn

Specifies the name of the MVS data set that is to be used as a model for allocation of new data sets. Specifying DCbdsn with no parameter value cancels the DCbdsn specification.

data_set_name

Specifies the name of the data set. The file name must be an MVS data set name. z/OS UNIX file names are not allowed on the DCbdsn parameter. The setting of QUOTESoverride is ignored. If the file name is enclosed in single quotation marks, it overrides the current working directory; otherwise it is appended to the current working directory.

Notes:

1. Specify the LOCSite parameters RECfm, LRecl, and BLKsize with no values to allow characteristics from the model DCB to be used.
2. To override the model characteristics of RECfm, LRecl, BLKsize, or RETpd, specify a value on the LOCSite command.
3. If MGmtclass is specified, the RETpd value of the MGmtclass can override the RETpd value.

Specifying a GDG data set with a relative index produces an error message. The following examples are unsupported specifications:

```
LOCSITE DCBDSN=MYGDG(0)
LOCSITE DCBDSN=MYGDG(-nnn) or
LOCSITE DCBDSN=MYGDG(+nnn)
```

See [“Steps for using a DCBDSN model to create a new data set” on page 84](#) for more information about DCbdsn.

Directory

Specifies the number of directory blocks to be allocated for the directory of a PDS.

Specify Directory=0 to allocate without specifying the number of directory blocks. Specify Directory=0 when you are also specifying DATAclass=*dataclass* and you want the SMS dataclass to provide the Directory size.

The size you specify with the Directory parameter overrides the DATAclass directory specification.

size

Specifies the number of directory blocks to be allocated for the directory of a PDS. The valid range is 1–16777215. The size 0 indicates that the directory blocks should be taken from the SMS data class.

DIRECTORYMode

Specifies that only the data set qualifier immediately below the current directory is treated as an entry in the directory. In directory mode, this data set qualifier is the only one used by the MPut subcommand.

DIRECTORYMode has no effect on files residing in a z/OS UNIX file system.

DSNTYPE

Specifies the data set name type for new physical sequential data sets.

SYSTEM

Physical sequential data sets are allocated with the SMS data class value. If no data class is defined, or if the DSNTYPE attribute is not defined, new physical sequential data sets will be allocated with the system default value.

BASIC

Allocates physical sequential data sets as physical sequential basic format data sets.

LARGE

Allocates physical sequential data sets as physical sequential large format data sets.

DSWAITTIME

Specifies the number of minutes that FTP waits when trying to access a local MVS data set.

minutes

The number of minutes to wait for a local MVS data set to become available. Valid values are in the range 0-14400. The value 0 (the default) specifies that FTP does not wait to obtain a data set when the data set is being held by another job.

EATTR

Specifies whether newly allocated data sets can have extended attributes and whether new data sets can reside in the EAS of an EAV.

SYSTEM

The data set uses the SMS data class EATTR value. If no SMS data class is defined, or if the data class contains no EATTR specification, the data set is allocated with the system default.

NO

The data set cannot reside in the EAS, and its VTOC entry cannot contain extended attributes.

OPT

The data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

ENCODING

Specifies the kind of encoding that is used for conversions between codepages for data transfers. See [“Support for SBCS languages” on page 77](#) and [“Support for MBCS languages” on page 81](#) for more information.

SBCS

Single Byte encoding. Code pages are specified using the SBDATACONN configuration option. This is the default value.

MBCS

Multibyte encoding. Code pages are specified using the MBDATACONN configuration option.

EPSV4

Specifies the client is to attempt to use the EPSV command to establish a data connection on an IPv4 session instead of referring to the FWFRIENDLY setting.

See RFC 2428 for information about the EPSV command. If the server rejects the EPSV command, the client refers to the FWFRIENDLY setting to determine how to establish the data connection. When the client is setting up proxy transfer data connections, it will try the EPSV and EPRT commands on IPv4 sessions; if a server rejects the EPSV or EPRT command, the client will try the PASV or PORT command instead.

If the server rejects either the EPSV or the EPRT command during the session, the client won't send EPSV to the server again, even when EPSV4 is specified.

FIFOIOTIME

Specifies the maximum length of time that the FTP client waits for an I/O operation to a named pipe in its z/OS UNIX file system to complete.

Rules:

- When you send a file that is a named pipe to the server, the FTP client reads from the named pipe one or more times. Each read from the named pipe must complete within the length of time that is specified by the FIFOIOTIME value.
- When you store a file that was received from the server as a UNIX named pipe, the FTP client writes to the named pipe one or more times. Each write to the named pipe must complete within the length of time that is specified by the FIFOIOTIME value.

seconds

The number of seconds that FTP waits for an I/O operation to a UNIX named pipe to complete. Valid values are in the range 1-86400. The default value is 20.

FIFOOPEN TIME

Specifies the length of time that the FTP client waits for an open of a named pipe in its z/OS UNIX file system to complete.

seconds

The number of seconds that FTP waits for an open of a named pipe to complete. Valid values are in the range 1-86400. The default value is 60.

FILEtype

Specifies the file type of the data set.

type

The file type of the data set can be:

Type

Description

SEQ

Sequential or partitioned data sets

SQL

SQL query function

FWFriendly

Specifies that the FTP client is firewall-friendly. This means that data connections will be set up from the FTP client to the FTP server.

Note: When the FTP server has an IPv6 address, data connections are always set up from the FTP client to the FTP server without reference to the FWFriendly setting.

ISPFSTATS

Allows FTP to create or update ISPF Member statistics when Get or MGet subcommands are issued.

LISTSUBdir

Use the LISTSUBdir option to indicate that wildcard searches should apply to the current working directory and should also span one level of its subdirectories. For the FTP client, this setting applies when issuing an MPut * subcommand.

Restriction: The LISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.

Result: If the LISTSUBdir option is not specified on the LOCSITE subcommand and the LISTSUBDIR statement is not specified in the client FTP.DATA file, the default is as if the LISTSUBdir option was specified on the LOCSITE subcommand.

LRecl

Used to specify the logical record length (LRecl) of a newly allocated data set.

Specify LRecl with no value when you are also specifying DATAclass=*data_class* and you want the SMS dataclass to provide the LRecl value, or when you are specifying DCbdsn=*data_set_name* and you want to use the LRecl from the DCBDSN data set. If LRecl=*length* is specified with either DATAclass or DCbdsn, the length specified by the LOCSite LRecl parameter overrides the DATAclass or DCbdsn LRecl.

length

Specifies the logical record length of a newly allocated data set. The valid range is 0 - 32760. A special value of x (LRecl=x) is also supported to indicate that a logical record length can be 32768 for variable-length spanned records.

Specifying LRecl=0 has the same effect as specifying LRecl with no parameters.

MBDATACONN=(*file_system_codepage,network_transfer_codepage*)

Specifies the codepages for the file system and for the network transfer used when the client does data conversion during a data transfer. This parameter affects the conversion of multibyte character set (MBCS) data (including support for DBCS code pages) and is used when the ENCODING=MBCS is also specified.

See [“Support for MBCS languages” on page 81](#) for more information.

file_system_codepage

Specifies the name of the file system codepage.

network_transfer_codepage

Specifies the name of the network transfer codepage.

MBREQUIRELSTEOL

Specifies that the FTP client will report an error when a multibyte file or data set is received from the server with no EOL sequence in the last record received. FTP will abort the file transfer.

MSENDEOL

Specifies which end-of-line sequence to use when ENCODING is MBCS, the data transfer type is ASCII, MODE is Stream, and data is being sent to the server. The following are possible values:

CRLF

Append both carriage return (X'0D') and line feed (X'0A') end-of-line sequences to each line of translated text. This is the default and the standard sequence defined by RFC 959. The z/OS server can receive ASCII data in this format only.

CR

Append only a carriage return (X'0D') end-of-line sequence to each line of translated text.

LF

Append only a line feed (X'0A') end-of-line sequence to each line of translated text.

NONE

Do not append an end-of-line sequence to the line of translated text.

Rules:

- Most servers support only the CRLF value for incoming ASCII data. Do not specify another value for MBSSENDEOL unless you have verified that the server is expecting the end-of-line sequence that you specify.
- Do not use an end-of-line sequence other than CRLF if the server is a z/OS FTP server. The z/OS FTP server supports only the CRLF value for incoming ASCII data.
- If you send a file to a server when MBSSENDEOL has a value other than CRLF, a subsequent SIZE command to that server targeting the file you sent could yield unpredictable results. Any size indicated in the server reply for such a file might not be reliable.

MGmtclass

Used to specify the SMS management class as defined by your organization for the target host. Specifying MGmtclass with no *mgmtclass* cancels the mgmtclass specification. The equal sign (=) is optional in this case.

mgmtclass

Specifies the SMS management class as defined by your organization for the target host. If the mgmtclass specified has a setting for RETpd, the value specified by the mgmtclass can override the setting of the LOCSite RETpd parameter, the RETpd value of a model data set if the DCbdsn parameter is specified, and the RETpd value defined in an SMS data class if DATAclass is specified. See [“Specifying values for new data sets” on page 82](#) for more information about specifying attributes when allocating new data sets.

MIGratevol

Specifies the volume ID for migrated data sets if they do not use IBM storage management systems. If you do not specify MIGratevol, the default *volume_serial* is MIGRAT.

volume_ID

The volume ID for migrated data.

NOASAtans

Treats ASA file transfers as regular file transfers; that is, the ASA characters are treated as part of the data and are not converted to print control characters.

NOAUTOMount

Prevents automatic mounting of volumes for data sets on volumes that are not mounted.

NOAUTOREcall

Prevents automatic recall of migrated data sets.

Note: A migrated data set can be deleted even though NOAUTOREcall is specified, because migrated data sets are not recalled for deletion.

NOCHKPTFLUSH

Specifies that z/OS can save the checkpoint records in the buffer and determine when to flush these records to the storage media.

NODBSUB

Specifies that substitution is not allowed for data bytes that cannot be translated in a double-byte character translation. This causes a data transfer failure if a character cannot be translated during the transfer. This is the default.

NOEPSV4

Prevents the client from using the EPSV command to establish a data connection on an IPv4 session. See RFC 2428 for information about the EPSV command. When NOEPSV4 is set, the client refers to the FWFRIENDLY setting to determine how to establish the data connection. When the client is setting up proxy transfer data connections, the client will use only PASV and PORT commands with IPv4 servers.

NOFWFriendly

Specifies that the FTP client is not firewall friendly. This means that data connections will be set up from the FTP server to the FTP client. This is the default behavior for FTP data connections.

Note: When the FTP server has an IPv6 address, data connections are always set up from the FTP client to the FTP server without reference to the FWFriendly setting.

NOISPFSTATS

Does not allow FTP to create or update ISPF Member statistics when Get or MGet subcommands are issued.

NOLISTSUBdir

Use the NOLISTSUBdir option to indicate that wildcard searches should apply only to the current working directory and should not span its subdirectories. For the FTP client, this setting applies when issuing an MPut * subcommand.

Restriction: The NOLISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.

Result: If the NOLISTSUBdir option is not specified on the LOCSITE subcommand and the LISTSUBDIR statement is not specified in the client FTP.DATA file, the default is as if the LISTSUBdir option was specified on the LOCSITE subcommand.

NOMBREQUIRELASTEOL

Specifies that the FTP client does not report an error when a multibyte file or data set is received from the network with no EOL sequence in the last record received. FTP will report the file transfer as completed.

NOPASSIVEIGNOREADDR

For passive mode FTP, specifies that the FTP client uses the IP address and port number from the PASV command reply that is returned by the FTP server for the data connection.

NOPASSIVEONLY

Specifies that the data connections for the client are not passive mode only. When passive mode attempt fails, the client tries active mode. This is the default.

NOQUOTEsoverride

A single quote at the beginning of the file name, as well as all other single quotation marks contained in the file name, is treated as part of the actual file name. The entire file name, including the leading single quotation mark, is appended to the current working directory.

NORDW

Specifies that variable record descriptor words (RDWs) are discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only.

NOREMOVEINBEof

Specifies that the UNIX end-of-file (EOF) byte (X'1A') is not removed on inbound ASCII transfers before the data is stored. See [z/OS Communications Server: IP Configuration Reference](#) for more information.

NORESTGet

Prevents opening the checkpoint data set for a Get request. Thus, checkpoint will not be active or recognized. Using this parameter when opening the checkpoint data set might cause a problem.

NOSBSUB

Specifies that substitution is not allowed for data bytes that cannot be translated in a single-byte character translation. This causes a data transfer failure if a character cannot be translated during the transfer.

NOSECUREIMPLICITzos

When the client connects to the server's TLS_{PORT}, the security handshake and negotiation are done immediately after the connect and before the 220 reply is received.

NOSPREad

Specifies that the output is in report format rather than spreadsheet format when the file type is SQL.

NOTRAILINGblanks

Specifies that the FTP client does not preserve the trailing blanks that are in a fixed format data set when the data is sent to a foreign host.

NOTRUNcate

Specifies that truncation is not permitted. The FTP client will set an error and fail file transfer if a record that is longer than LRECL of the new file is detected.

Note: If WRAPRECORD is set then the data is wrapped, not truncated, no error will be set, and the file transfer will continue.

NOUCSSUB

In UCS-2-to-EBCDIC conversion, the data transfer is terminated if any UCS-2 character cannot be converted into the EBCDIC code set.

NOUCSTRUNC

In UCS-2-to-EBCDIC conversion, truncation of EBCDIC data is not allowed. The data transfer is aborted if the logical record length of the receiving data set is too small to contain the data after conversion to EBCDIC.

Note: The setting of the CONDdisp parameter determines what happens to the target data set if the transfer is aborted.

NOWRAPrecord

Indicates that data is truncated if no new line character is encountered before the logical record length of the receiving file is reached.

Note: If NOTRUNcate is also set, an error will be set and the file transfer will fail.

NOWRTAPEFastio

Specifies that ASCII stream data that is being written to tape must be written using the Language Environment® run time library.

PASSIVEIGNOREADDR

For passive mode FTP, specifies that the FTP client uses the port number from the PASV command reply and the IP address that was used to log in to the FTP server, for the data connection.

Requirement: See the [PASSIVEIGNOREADDR \(FTP client\) information in z/OS Communications Server: IP Configuration Reference](#) for more details about the requirements.

PASSIVEONLY

Specifies that the data connections for the client are passive mode only.

PDSTYPE

Specifies whether the FTP client creates local MVS directories as partitioned data sets or as partitioned data sets extended.

When specified without a value, FTP will not specify to z/OS whether to allocate a new MVS directory as a PDS or a PDSE. When specified without a value, the equal sign (=) is optional.

PDS

Allocate directories as partitioned data sets.

PDSE

Allocate directories as partitioned data sets extended.

PROGRESS

Specifies the interval between progress report messages generated by the FTP client during a file transfer (inbound or outbound).

number

Specifies the interval (in seconds) between progress report messages that are generated in the FTP client during an inbound or outbound file transfer. Valid values are in the range 10-86400, or 0. The value 0 turns off progress reporting in the FTP client. The default value is 10 seconds.

The messages that are generated as part of progress reporting are EZA2509I and EZA1485I. These messages are generated automatically at 10-second intervals by the FTP client in releases prior to version V1R6. Beginning in version V1R6, the default behavior is the same as in prior releases, but the length of the interval and whether to generate the messages can be configured by using the PROGRESS parameter setting on the LOCSITE subcommand or by specifying the PROGRESS statement in the FTP.DATA file.

PRImary

Used to specify the number of tracks, blocks, or cylinders for primary allocation. When specified with a value of 0, no primary value is used when allocating the data set.

Specify a PRImary allocation of 0 when you are also specifying DATAClass=*data_class* and when you want the SMS dataclass to provide the PRImary *amount*.

To enable the SMS data class to determine the space allocation, both PRImary and SEConDary allocations must be specified as 0. The tracks, blocks, cylinders setting is ignored in this case. If PRImary with *amount* not equal to 0 is specified with DATAClass, the value specified by the LOCSite PRImary parameter overrides the DATAClass space allocation.

amount

Specifies the number of tracks, blocks, or cylinders for primary allocation. For allocating partitioned data sets, this is the amount of space that is allocated for the primary extent.

For allocating sequential data sets this is the maximum amount of space that is allocated for the primary extent. If a lesser amount of space is needed to hold the data being transferred, only the amount of space actually needed to hold the data is allocated. The valid range is 1–16777215.

Qdisk

Used to display statistics about the amount of space available on a volume. If Qdisk is entered without a specific *volume_serial*, statistics about available space are displayed for each volume that is defined with “Use Attribute=storage”.

volume_serial

Displays statistics about available space on a specific volume.

QUOTESoverride

Specifies that a single quotation mark at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory. This is the way single quotation marks are used in all previous MVS FTP servers, and this is the default. Any single quotation mark inside the beginning and ending quotation marks are treated as part of the file name.

QUOTESoverride indicates the usage of single quotation marks appearing at the beginning of, or surrounding, a file name. The setting of this keyword affects all FTP subcommands that have a path name as a parameter except keywords on the LOCSite subcommand.

RDW

Specifies that variable record descriptor words (RDWs) are treated as if they were part of the record and are not discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only.

Note: RDW information is stored in binary format. Transfer files in binary mode to avoid the translation problems that can occur if you transfer this binary field in EBCDIC or ASCII mode.

READTAPEFormat

Used to provide information about an input data set on tape. If specified without the *tape_format* (which is the default), processing of input tapes does not take advantage of the record format information prior to open. The equal sign (=) is optional in this case.

The READTAPEFormat parameter has no effect on, and is not affected by DATAClass, DCbdsn, LRecl, RECfm, or any other parameters associated with creating a data set.

tape_format

Specifies the format of the records on the input tape data set. Valid formats are:

- F**
Fixed record length
- V**
Variable record length
- S**
Spanned records
- X**
Logical record length is undefined (Lrecl X)

blank

Unspecified (displayed as U in messages and reply)

These formats are mutually exclusive. Spanned implies variable, and Lrecl X implies spanned. If specified, the *tape_format* value must be the most inclusive identifier in the list that matches the tape. If it is not the most inclusive identifier, an error message is issued. For example, if the *tape_format* value is **S** (spanned) and the tape contains records with undefined length (Lrecl X), the request will fail. An unspecified format avoids this type of error. However, the following should be considered:

- Specify a value for the READTAPEFormat parameter in all the following cases. Failure to specify a format will likely cause errors in processing the tape.
 - The record length is undefined (Lrecl X).
 - The records are spanned (Recfm is VBS, VS).
 - The records are variable (Recfm is V, VB, VBA) and RDW is specified.
- Specify a value for the READTAPEFormat parameter for all input tapes that have one of the listed formats to ensure best results.

RECfm

Used to specify the record format of a data set. When specified without the *format*, no record format is used when allocating the data set. The equal sign (=) is optional in this case.

Specify the RECfm parameter with no value when you are also specifying DATAClass=*data_class* and you want the SMS dataclass to provide the RECfm *format*, or when you are specifying DCbdsn=*data_set_name* and you want to use the record format from the DCBDSN data set.

If RECfm=*format* is specified with either DATAClass or DCbdsn, the value specified by the LOCSItE RECfm parameter overrides the DATAClass or DCbdsn record format.

format

Specifies the record format of a data set. Valid record formats are: F, FA, FB, FBA, FBM, FBS, FBSA, FBSM, FM, FS, FSA, FSM, U, UA, UM, V, VA, VB, VBA, VBM, VBS, VBSA, VBSM, VM, VS, VSA, and VSM. The characters used to specify these record formats have the following meanings:

| Code | Description |
|-------------|---|
| F | Fixed record length |
| V | Variable record length |
| U | Undefined record length |
| B | Blocked records |
| S | Spanned records (if variable) / standard records (if fixed) |
| A | Records contain ISO/ANSI control characters |

| Code | Description |
|------|---|
| M | Records contain machine code control characters |

REMOVEINBOF

Specifies that the UNIX end-of-file (EOF) byte (X'1A') is removed on inbound ASCII transfers before the data is stored. See [z/OS Communications Server: IP Configuration Reference](#) for more information.

RESTGet

Allows opening the checkpoint data set for the Get request. This is the default when the RESTGet statement has not been added to the FTP.DATA file.

RETPd

Used to specify the number of days that a newly allocated data set should be retained.

Specify RETpd with no value when you are also specifying `DATAclass=data_class` or `MGmtclass=mgmtclass` and you want SMS to provide the RETpd value, or when you are specifying `DCbdsn=data_set_name` and you want to use the RETpd from the DCBDSN data set. If more than one of the LOCSite parameters (RETPd, MGmtclass, DATAclass, or DCbdsn) are specified, the order of precedence (highest to lowest) is:

1. MGmtclass
2. RETpd
3. DCbdsn
4. DATAclass

If a retention period is associated with an SMS management or data class, or with a model DCBDSN data set, the value of the retention period can be overridden to another nonzero value, but it cannot be overridden to have no retention period specified for the newly created data sets.

days

Specifies the number of days that a newly allocated data set should be retained. The valid range is 0–9999. A value of 0 indicates a retention period of 0 days so that the data set expires the same day as it was created.

SBDataconn

Specifies the conversions between file system and network code pages to be used for data transfers.

```
LOCSITE SBDataconn=dsname
LOCSITE SBDataconn=(file_system_cp,network_transfer_cp)
LOCSITE SBDataconn=FTP_STANDARD_TABLE
LOCSITE SBDataconn=*
LOCSITE SBDataconn=
LOCSITE SBDataconn
```

See [“Support for SBCS languages” on page 77](#) for more information.

The following forms of specifying SBDataconn are equivalent to specifying SBDataconn=*

- SBDataconn
- SBDataconn=

dsname

Specifies the fully qualified name of an MVS data set or z/OS UNIX file that contains the EBCDIC-to-ASCII and ASCII-to-EBCDIC translate tables generated by the CONVXLAT utility.

Notes:

1. The name must *not* be enclosed in quotation marks. If quotation marks appear, they are treated as part of the name. (QUotesoverride is ignored.)
2. The z/OS UNIX file system name is case sensitive. The MVS name is not case sensitive.
3. The name cannot begin with a left parenthesis ([).

4. The SBDataconn keyword must be the only keyword or the last keyword on a LOCStite subcommand.

file_system_cp

Specifies the name of the file system code page recognized by iconv. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

network_transfer_cp

Specifies the network transfer code page recognized by iconv. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

FTP_STANDARD_TABLE

Specifies that the FTP internal tables, which are the same as the tables that are shipped in TCPXLBIN(STANDARD), are to be used on the data connection.

Specifies the translate tables set up at initialization for the data connection must be used.

SBSENDEOL

Specifies which end-of-line sequence to use when ENCODING is SBCS, the data transfer type is ASCII, and data is being sent to the server. The following are possible values:

CRLF

Append both carriage return (X'0D') and line feed (X'0A') end-of-line sequences to each line of translated text. This is the default and the standard sequence defined by RFC 959. The z/OS server can receive ASCII data in this format only.

CR

Append only a carriage return (X'0D') end-of-line sequence to each line of translated text.

LF

Append only a line feed (X'0A') end-of-line sequence to each line of translated text.

NONE

Do not append an end-of-line sequence to the line of translated text.

Tip: The **srestart** subcommand is disabled if you configure an SBSENDEOL value other than CRLF.

Rules:

- Most servers support only the CRLF value for incoming ASCII data. Do not specify another value for SBSENDEOL unless you have verified that the server is expecting the end-of-line sequence that you specify.
- Do not use an end-of-line sequence other than CRLF if the server is a z/OS FTP server. The z/OS FTP server supports only the CRLF value for incoming data.
- If you send a file to a server while SBSENDEOL has a value other than CRLF, a subsequent SIZE command to that server targeting the file you sent could yield unpredictable results. Any size indicated in the server reply for such a file might not be reliable.

SBSUB

Specifies that substitution is allowed for data bytes that cannot be translated in a single byte character translation. The substitution character is specified by the SBSUBCHAR parameter.

SBSUBCHAR nn

Specifies the value that is used for substitution when SBSUB is also specified. The value is one of the following:

SPACE

When the target code set is ASCII, replace untranslatable characters with X'20' during SBCS data transfers. When the target code set is EBCDIC, replace untranslatable characters with X'40' during SBCS data transfers.

nn

Replace untranslatable characters with *nn* during SBCS data transfers, where *nn* is a hexadecimal value in the range 00-FF.

SECondary

Specifies the amount of tracks, blocks, or cylinders for secondary allocation.

Specify SECondary=0 when you are also specifying DATAclass=*dataclass* and you want the SMS dataclass to provide the SECondary value. To enable the SMS data class to determine the space allocation, both PRIMary and SECondary must be specified as 0. The tracks, blocks, cylinders setting is ignored in this case. If SECondary is specified as other than 0 with DATAclass, the value specified by the Site SECondary parameter overrides the DATAclass space allocation.

amount

Specifies the number of tracks, blocks, or cylinders for secondary allocation. The valid range is 0–16777215. If you specify the *amount* value 0, FTP allocates without specifying secondary space.

SECUREIMPLICITzos

When the client connects using the TLSPORT implicit connection, the client waits for the 220 *good morning* reply before initiating the security handshake and negotiation. This is the default.

SPRead

Specifies that the output is in spreadsheet format when the file type is SQL.

SQLCol

Specifies the column headings of the SQL output file.

any

The label of the Db2 SQL table column heading is the first choice for column heading, but if there is no label, the name becomes the column heading.

labels

Labels are the Db2 SQL table column headings. If any of the columns do not have labels, FTP supplies a column heading in the form of COLnnn.

names

Uses the names of the Db2 SQL table column headings. The labels are ignored.

STOrclass

Specifies the SMS storage class as defined by your organization for the target host. Cancels the storage class specification when specified without a *storage_class* parameter value. The equal sign (=) is optional in this case.

See “[Specifying values for new data sets](#)” on page 82 for more information about specifying attributes when allocating new data sets.

storage_class

Specifies the SMS storage class as defined by your organization for the target host.

When an SMS storage class is in use, any of the attributes specified there can be overridden by a different specification by the user. To avoid overriding the setting in the SMS storage class, specify BLKSize, LRecl, PDSTYPE, PRIMary, RECFm, SECondary, UCOUNT, Unit, VCOUNT, or VOLUME with no associated value. This removes any value specified on a prior LOCSITE command or in FTP.DATA, and the affected attributes are not included on the allocation. To override a setting in the SMS storage class, specify the wanted value with the appropriate keyword.

TLSRFCLEVEL

Specifies the level of RFC 4217 that the client supports for TLS-protected sessions. See “[Using security mechanisms](#)” on page 39 for more information.

DRAFT

The Internet draft revision level of RFC 4217 is supported. This is the level of RFC 4217 support the z/OS FTP client has offered since CSV1R2. This is the default value.

RFC4217

The FTP client complies with RFC 4217.

CCCNONOTIFY

The FTP client does not issue the TLSshutdown command after sending the CCC command. RFC 4217 did not mandate this flow until Internet revision 14.

Result: The SECUREIMPLICITZOS configuration option is not affected by this setting.

Note: FTP supports the TLS`SPORT` statement regardless of the TLS`RFCLEVEL` setting. FTP connections to the TLS`SPORT` port are implicitly secured with TLS as described in the Internet draft.

Restrictions:

- You cannot set the TLS`RFCLEVEL` setting to DRAFT during a TLS-secured session when the control connection is not secured.
- When you set the TLS`RFCLEVEL` option to RFC4217 or CCC`NONOTIFY`, the FTP server must use the same setting. If the server setting does not match, the connection might be reset or the session might hang and eventually time out.
- The CCC`NONOTIFY` option is not valid with the TLS`MECHANISM` ATTLS option. If both options are specified, using the CCC command causes the FTP session to fail. If CCC`NONOTIFY` is required for the partner system, configure TLS`MECHANISM` FTP with associated statements and exemption in the TTLSRules.

TRacks

Specifies that primary and secondary space allocations are in tracks.

If both PRI`mary` and SE`condary` are specified as 0 and an SMS data class has been specified, the space allocation is determined by the SMS data class and the TRacks parameter is ignored.

TRAILingblanks

Specifies that the FTP client preserves the trailing blanks in a fixed format data set when the data is sent to a foreign host.

TRUNcate

Specifies that truncation is permitted. The FTP client does not set an error when a truncated record is detected and the file transfer continues.

UCOUNT

Specifies how many devices to allocate concurrently to support the allocation request.

unit_count

Specifies number of devices to allocate. Valid value is 1–59. When specified without a value, the FTP server does not specify a unit count when allocating data sets.

P

Parallel mount request.

Guideline: The UCOUNT statement is not meant to be used with an SMS storage class. Any UCOUNT value you specify overrides whatever is specified for the SMS managed dataclass being used.

UCSHOSTCS

Specifies the EBCDIC code set to be used when converting to and from UCS-2. If you do not specify a *code_set* value, the current code set is used.

code_set

Name of the EBCDIC code set to be used when converting to and from UCS-2.

UCSSUB

In UCS-2-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any UCS-2 character that cannot successfully be converted. Data transfer continues.

UCSTRUNC

In UCS-2-to-EBCDIC conversion, truncation of EBCDIC data is allowed. The data transfer continues even if EBCDIC data is truncated.

UMask

Defines the file mode creation mask. The file mode creation mask defines which permission bits are *not* to be set on when a file is created. When a file is created, the permission bits requested by the file creation are compared to the file mode creation mask, and any bits requested by the file creation which are not allowed by the file mode creation mask are turned off.

The format of the UMask keyword is UMask *ooo*.

When a file is created, the specified permission bits for the file are 666 (-rw-rw-rw-). If the file mode creation mask is 027, the requested permissions and the file mode creation mask are compared:

```
110110110 - 666
000010111 - 027
-----
110100000 - 640
```

The actual permission bits set for the file when it is created is 640 (-rw-r-----).

Notes:

1. The default value for UMask is 027.
2. You cannot use FTP to create z/OS UNIX files with execute permissions. If you require execute permissions, use the LOCSite CHMod command to change permissions after the file has been created.

UNICODEFILESYSTEMBOM

Specifies whether the FTP client stores incoming Unicode files with a byte order mark (BOM).

Restriction: The only Unicode encoding formats supported for file storage by z/OS FTP are UTF-8 and UTF-16. Files are always stored in big endian format.

Result: The byte order mark (BOM) stored with the file is determined by the encoding used to store the file rather than by the format of the BOM sent with the file.

ASIS

Store incoming Unicode files with a byte order mark only if the file was sent with a byte order mark

ALWAYS

Store incoming Unicode files with a byte order mark regardless of whether the file was sent with a byte order mark.

NEVER

Store incoming Unicode files without a byte order mark regardless of whether the file was sent with a byte order mark.

Results:

- The Unicode byte order mark, U+FEFF, can also be interpreted as a zero-width nonbreaking space character. z/OS FTP considers only the first character of the data received from the server as a possible byte order mark (BOM). No other instance of the BOM sequence in the inbound data is affected by this setting.
- When the local file is a z/OS UNIX named pipe, incoming data is always appended to any existing data that is in the named pipe. If you code UNICODEFILESYSTEMBOM = ASIS or ALWAYS and the named pipe contains data, the client appends a BOM byte sequence to existing data in cases in which it would add a BOM at the beginning of a regular file. The BOM byte sequence is interpreted as a zero-width nonbreaking space character when it does not start the file or data stream. You must take this into consideration when you configure the UNICODEFILESYSTEMBOM parameter.

Unit

Specifies the unit type for allocation of new data sets.

unit_type

The unit type (for example, 3380) for the allocation of new data sets on direct access devices. If *unit_type* is not specified, the unit type used for allocation is set back to the system default.

UNIXFILETYPE

Specifies whether to treat files in the z/OS UNIX file system as regular files or as UNIX named pipes.

FILE

Treat files in the z/OS UNIX file system as regular files. This is the default.

FIFO

Treat files in the z/OS UNIX file system as UNIX named pipes.

For information about transferring data into and from z/OS UNIX named pipes, see [“Using z/OS UNIX System Services named pipes”](#) on page 104.

VCOUNT

Specifies the number of tape data set volumes that an allocated data set can span. When this parameter is specified without a *volume_count* value, the FTP server uses the volume count 50 when it allocates tape data sets.

volume_count

Valid values are in the range 1-255.

VOLUME

Specifies the volume serial number for allocation of new data sets.

volume_serial

The serial number of the volume to use for allocation.

volume_serial_list

A list of one or more volume serial numbers for allocation. Delimit each volume serial number from the previous one with a comma.

If VOLUME is specified without a *volume_serial_list* or *volume_serial* parameter, no volumes are specified by the FTP client during the allocation of a new data set, and the installation default is used.

WRAPRECORD

Specifies that data is wrapped to the next record if no new line character is encountered before the logical record length of the receiving file is reached.

WRTAPEFASTIO

Specifies that ASCII Stream data that is being written to tape is allowed to be written using BSAM I/O.

Tips:

- You can specify more than one parameter with the LOCSite subcommand. Delimit each parameter with a blank space.
- Issue the HELP LOCSite subcommand to display a list of configuration options available on the local host.

Results:

- The site-dependent information set with the LOCSite subcommand remains active until you issue a new LOCSite subcommand. The new LOCSite subcommand adds to or changes the parameters established by previous LOCSite subcommands.
- If you specify one or more incorrect parameters with the LOCSite subcommand, an error message specifying the incorrect parameter is displayed. All correct parameters are set, regardless of any incorrect parameters, and do not need to be reissued.

Related topics:

- See [“HELP and ? subcommands—Display help information”](#) on page 171 for more information about the HELP subcommand.
- To check the effect of the LOCSite command on the attributes at the local host, see [“LOCStat subcommand—Display local status information”](#) on page 201.

LOCStat subcommand—Display local status information

Purpose

Use the LOCStat subcommand to display local status information.

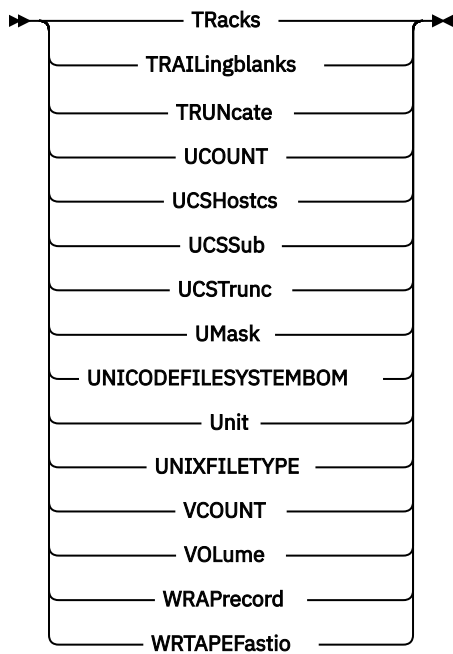
Tip: Issuing the LOCStat subcommand with no parameters causes all local status to be displayed. To display local status for a single configuration option, issue the LOCStat subcommand with at least one parameter.

Format

►► LOCStat — option ◄◄

options

| |
|-------------------|
| ASATrans |
| AUTOMount |
| AUTORECALL |
| BLOCKS |
| BLOCKSIZE |
| BUfno |
| CConntime |
| CHKPtint |
| CHKPTPrefix |
| CONDdisp |
| Cylinders |
| DATAclass |
| DATActime |
| DATAKEEPALIVE |
| DATASetmode |
| DB2 |
| DBSUB |
| DCbsdn |
| DCOnntime |
| Directory |
| DIRECTORYMode |
| DSNTYPE |
| DSWAITTIME |
| EATTR |
| ENcoding |
| EPSV4 |
| FIFOIOTIME |
| FIFOOPENINGTIME |
| FILEtype |
| FTPkeepalive |
| FWFriendly |
| INacttime |
| ISPFStats |
| LISTSUBdir |
| LRecl |
| MBdataconn |
| MBREQUIRELASTEOL |
| MBSENDEOL |
| MGmtclass |
| MIGratevol |
| MYopentime |
| PASSIVEIGNOREADDR |
| PASSIVEONLY |
| PDSTYPE |
| PRImary |
| QUOTESoverride |
| RDw |
| READTAPEFormat |
| RECFm |
| RESTGet |
| RETPd |
| SBDdataconn |
| SBSSENDEOL |
| SBSUB |
| SBSUBChar |
| SECOndary |
| SECUREIMPLICITZOS |
| SPRead |
| SQLCol |
| STORclass |
| TLSRFCLEVEL |



Parameters

ASAtans

Indicates that the FTP client interprets characters in the first column of ASA files that are being transferred as print control characters.

AUTOMount

Indicates automatic mounting of volumes for data sets that are on unmounted volumes.

AUTOREcall

Indicates automatic recall of migrated data sets.

BLOCKS

Indicates that primary and secondary space allocations are in blocks.

BLOCKSize

Indicates the block size of a newly allocated data set.

BUfno

Indicates the number of access method buffers that is used when data is read from or written to a data set.

CConntime

Indicates the length of time that the FTP client waits after attempting to close a control connection before terminating it and reporting an error.

CHKptint

Indicates the checkpoint interval for the sending site in a file transfer request.

CHKPTPrefix

Indicates a key that is used to determine the high-level qualifier of the checkpoint data set.

CONDdisp

Indicates the disposition of the data set if a retrieve operation for a new data set ends before all of the data is written.

CYLinders

Indicates that primary and secondary space allocations are in cylinders.

DATAclass

Indicates the SMS data class.

DATAKEEPAIVE

Indicates the number of seconds that TCP/IP waits before sending a keepalive packet while the data connection is inactive.

The value 0 indicates that the DATAKEEPAIVE timer is disabled. For passive mode data connections, the keepalive timer that you configured in PROFILE.TCPIP controls how often keepalive packets flow on the data connection. For active mode data connections, FTP suppresses the PROFILE.TCPIP keepalive timer.

DATActime

Indicates the length of time that the FTP client waits after attempting to send or receive data before terminating the connection and reporting an error to the user.

DATASetmode

Indicates whether DATASetmode or DIRECTORYMode is in effect.

DB2

Indicates the name of the Db2 subsystem.

DBSUB

Indicates that substitution is allowed for data bytes that cannot be translated in a double-byte character translation.

DCbdsn

Indicates the name of the MVS data set to be used as a model for allocating new data sets.

DConntime

Indicates the amount of time that FTP waits attempting to close a data transfer before terminating the connection and reporting an error.

Directory

Indicates the number of directory blocks to be allocated for the directory of a partitioned data set (PDS).

DIRECTORYMode

Indicates whether DATASetmode or DIRECTORYMode is in effect.

DSNTYPE

Indicates the data set name type for new physical sequential data sets.

SYSTEM

Physical sequential data sets are allocated with the SMS data class value. If no data class is defined, or if the DSNTYPE attribute is not defined, new physical sequential data sets will be allocated with the system default value.

BASIC

Allocates physical sequential data sets as physical sequential basic format data sets.

LARGE

Allocates physical sequential data sets as physical sequential large format data sets.

DSWAITTIME

Indicates the number of minutes the FTP client waits for an MVS data set to become available when a local data set is held by another job or process. The value 0 indicates that the FTP client does not wait to obtain a data set when the data set is held by another job or process.

EATTR

Indicates whether newly allocated data sets can have extended attributes and whether new data sets can reside in the EAS of an EAV. The value of EATTR is one of the following values:

SYSTEM

The data set uses the SMS data class EATTR value. If no SMS data class is defined, or if the data class contains no EATTR specification, the data set is allocated with the system default.

NO

The data set cannot reside in the EAS, and its VTOC entry cannot contain extended attributes.

OPT

The data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

ENCODING

Indicates the kind of encoding that is used for conversions between code pages for data transfers.

EPSV4

Indicates that the client will attempt to use the EPSV command to establish a data connection on an IPv4 session instead of referring to the FWFRIENDLY setting.

FIFOIOTIME

Indicates the length of time that the FTP client waits for a read from a z/OS UNIX named pipe or for a write to a z/OS UNIX named pipe to complete.

FIFOOPENTIME

Indicates the length of time that the FTP client waits for an open of a z/OS UNIX named pipe to complete.

FILEtype

Indicates the file type of the data set.

FTpkeepalive

Indicates the control connection keepalive timer value in seconds.

FWFriendly

Indicates that the FTP client is firewall-friendly.

INacttime

Indicates the length of time that the FTP client waits for an expected response from the server, on either the control or the data connection, before closing the session.

ISPFSTATS

Indicates whether FTP will create or update ISPF Member statistics when Get or MGet subcommands are issued.

LISTSUBdir

Indicates that wildcard searches should apply to the current working directory and should also span its subdirectories.

LRecl

Indicates the logical record length (LRecl) of a newly allocated data set.

MBDATACONN

Indicates the code pages for the file system and for the network transfer that are used when the client does data conversion during a data transfer.

MBREQUIRELASTEOL

Indicates whether the last record of an incoming multibyte transfer is required to have an EOL sequence.

TRUE

A missing EOL on the last record received is treated as an error.

FALSE

A missing EOL on the last record received is ignored.

MBSSENDEOL

Indicates which end-of-line sequence to use when the ENCODING value is SBCS, the data is ASCII, and data is being sent to the server.

MGmtclass

Indicates the SMS management class.

MIGratevol

Indicates the volume ID for migrated data sets if they do not use IBM storage management systems.

MYopentime

Indicates the length of time that the FTP client waits for a session to open before terminating the attempt and reporting an error.

PASSIVEIGNOREADDR

Indicates whether the FTP client should ignore the IP address in the FTP server PASV reply for the data connection and use the IP address that was used to log in to the FTP server.

PASSIVEONLY

Indicates whether the data connections for the client are passive mode only.

PDSTYPE

Indicates whether the FTP client creates local MVS directories as partitioned data sets or as partitioned data sets extended.

PRImary

Indicates the number of tracks, blocks, or cylinders for the primary allocation.

QUOTESoverride

Indicates that a single quotation mark (') at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory.

RDW

Indicates that variable record descriptor words (RDWs) are treated as if they are part of the record and are not discarded during FTP transmission of variable format data sets.

READTAPEFormat

Indicates information about an input data set on tape.

RECFm

Indicates the data set record format.

RESTGet

Indicates that opening the checkpoint data set for the Get request is allowed.

RETpd

Indicates the number of days that a newly allocated data set should be retained.

SBDatconn

Indicates the conversions between file system and network code pages to be used for data transfers.

SBSSENDOL

Indicates which end-of-line sequence to use when ENCODING is SBCS, the data is ASCII, and data is being sent to the client.

SBSUB

Indicates that substitution is allowed for data bytes that cannot be translated in a single byte character translation.

SBSUBCHAR

Indicates the value that is used for substitution when SBSUB is also specified.

SECondary

Indicates the number of tracks, blocks, or cylinders for a secondary allocation.

SECUREIMPLICITzos

Indicates that when the client connects using the TLS_{PORT} implicit connection, the client waits for the 220 good morning reply before initiating the security handshake and negotiation.

SPRead

Indicates that the output is in spreadsheet format when the file type is SQL.

SQLCol

Indicates the SQL output file column headings.

STOrclass

Indicates the SMS storage class.

TLSRFCLEVEL

Indicates the level of RFC 4217, *On Securing FTP with TLS*, that is supported by the client.

TRacks

Indicates that primary and secondary space allocations are in tracks.

TRAILingblanks

Indicates that the FTP client preserves the trailing blanks in a fixed-format data set when the data is sent to a foreign host.

TRUNcate

Indicates that truncation is permitted.

UCOUNT

Indicates how many devices to allocate concurrently to support the allocation request.

UCSHOSTCS

Indicates the EBCDIC code set to be used when converting to and from Unicode.

UCSSUB

Indicates that in Unicode-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any Unicode character that cannot be successfully converted.

UCSTRUNC

Indicates that in Unicode-to-EBCDIC conversion, EBCDIC data truncation is allowed.

UMask

Indicates the file mode creation mask.

UNICODEFILESYSTEMBOM

Indicates whether the FTP client stores incoming Unicode files with a byte order mark.

UNIXFILETYPE

Indicates whether the FTP client treats files in its z/OS UNIX file system as regular files or as z/OS UNIX named pipes.

Unit

Indicates the unit type for allocation of new data sets.

VCOUNT

Indicates the number of tape data set volumes that an allocated data set can span.

VOLUME

Indicates the volume serial number for allocation of new data sets.

WRAPrecord

Indicates that data is wrapped to the next record if no new-line character is encountered before the logical record length of the receiving file is reached.

WRTAPEFastio

Indicates that ASCII stream data that is being written to tape can be written using BSAM I/O.

Examples

The following is an example of the output from the LOCSTat subcommand, issued with a single parameter:

```
Command: locstat dconntime
DCONNTIME is 120
Command:
```

The following example shows part of the output from a LOCSTat subcommand.

```

Command: locstat
Trace: FALSE, Send Port: TRUE
Send Site with Put command: TRUE
Connected to:9.42.104.38, Port: FTP control (21), logged in
local site variable DSWAITTIME is set to 0
VCOUNT is 59
Prompting: ON, Globbing: ON
ASA control characters transferred as ASA control characters
New data sets catalogued if a store operation terminates abnormally
Single quotes will override the current working directory
UMASK value is 027
Data connections for the client are not firewall friendly.
local site variable EPSV4 is set to TRUE
local site variable SECUREIMPLICITZOS is set to TRUE
local site variable PASSIVEIGNOREADDR is set to FALSE
local site variable TLSRFCLEVEL is set to RFC4217
local site variable READVB is set to LE
local site variable EXTDBSCHINESE is set to TRUE
local site variable LISTSUBdir is set to TRUE
local site variable PROGRESS is set to 10
local site variable SEQNUMSUPPORT is set to FALSE
local site variable UNIXFILETYPE is set to FILE
local site variable FIFOIOTIME is set to 20
local site variable FIFOOPTIME is set to 60
local site variable EATTR is set to SYSTEM
local site variable DSNTYPE is set to BASIC
Authentication mechanism: None Local Port: 1118
Tape write is not allowed to use BSAM I/O
Using /etc/ftp.data for local site configuration parameters.
Command:

```

LPwd subcommand—Display the current working-level qualifier

Purpose

Use the LPwd subcommand to display the name of the current working directory on the local host.

Format

➤ LPwd ➤

Parameters

There are no parameters for this subcommand.

Examples

Display the name of the current working directory:

```

lpwd
Local directory is partitioned data set USER14.FTP.TEST1.
Command:

```

LS subcommand—Obtain a list of file names

Purpose

Use the LS subcommand to list only the names of a set of remote files, file group, or directory.

Format

➤ LS ———— ➤
 └── name ─┘ └── (— Disk —) ─┘

Parameters

name

Specifies the set of remote files whose names are to be listed. The default is the entire current directory or file group.

Disk

Stores the results of the LS subcommand in the *user_id*.FTP.LSOUTPUT data set. The results are not displayed on the screen.

Note: If the local current working directory is a z/OS UNIX file system directory, the results are stored in a file named LSOUTPUT.

Restriction: The LISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.

Results:

- If the LISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the server FTP.DATA file, the default is as if the LISTSUBdir option was specified on the SITE subcommand.
- If the z/OS FTP server has the NOLISTSUBdir option specified on the SITE subcommand or has LISTSUBDIR FALSE specified in the server FTP.DATA file, an ls * command will list only the files in the current directory.

Examples

Example 1: The following is a sample response that displays after using the LS subcommand.

```
>>>PORT 9,67,58,227,4,63
200 Port request OK.
>>>NLST
125 List started OK.
A.X
CHR.TXT
OBEY.TCPIP
PROFILE.EXEC
SPF.ISPPROF
USERTRAN.TCPXLBIN
250 List completed successfully.
Command:
```

The following is a sample entry and the response that displays after using the LS subcommand listing z/OS UNIX files.

```
cd '/u/user121/ftp.example'

>>>CWD '/u/user121/ftp.example'
250 HFS directory /u/user121/ftp.example is the current working directory
Command:
ls
>>>PORT 9,67,112,25,4,62
200 Port request OK.
>>>NLST
125 List started OK
append02
file1
file2
file3
file4
file5
250 List completed successfully.
Command:
```

Example 2: ls * with SITE LISTSUBdir

Following is an example of ls * with SITE LISTSUBdir. This setting affects processing of the NLST command. The z/OS FTP client sends an NLST command to the server as part of ls * subcommand processing. The LISTSUBdir option specifies that not only the current subdirectory, but also the next

subdirectory should be listed as a result of processing an `ls *` subcommand. In this example, the current directory has a file `x` and a subdirectory `y` and subdirectory `y` has a file `x`.

```
site listsubdir
>>> SITE listsubdir
200 SITE command was accepted
ls *
>>> PORT 127,0,0,1,4,17
200 Port request OK.
>>> NLST *
125 List started OK
x
y/x
250 List completed successfully.
Command:
```

Example 3: `ls *` with `SITE NOLISTSUBdir`

Following is an example of `ls *` with `SITE NOLISTSUBdir`. This setting affects processing of the `NLST` command. The z/OS FTP client sends an `NLST` command to the server as part of `ls *` subcommand processing. The `NOLISTSUBdir` option specifies that only the current directory should be listed as a result of processing an `ls *` subcommand. In this example, the current directory has a file `x` and a subdirectory `y` and subdirectory `y` has a file `x`.

```
site Nolistsubdir
>>> SITE Nolistsubdir
200 SITE command was accepted
ls *
>>> PORT 127,0,0,1,4,18
200 Port request OK.
>>> NLST
125 List started OK
x
y
250 List completed successfully.
Command:
```

Usage

- To make a file group the current working directory, use the `CD` subcommand. The method you use to specify a directory or file group is host-dependent.
- You can use special characters for pattern matching when specifying the *name*. These characters depend on the host FTP server. See [“DlR subcommand—Obtain a list of directory entries” on page 159](#) for information about using special characters with the z/OS FTP server.
- If the current local directory is a PDS, only a member named `LSOUTPUT` is created. If the current local directory is not a PDS, the local directory, not the user ID, is used as the high-level qualifier for the data set name.
- If the local current working directory is a z/OS UNIX file system directory, the results are stored in a file named `LSOUTPUT`.
- `LS` lists entries only for data sets and file types that FTP can process (see [Appendix A, “Specifying data sets and files,” on page 359](#) for a list). GDG base, VSAM, and ATL library entries are among the types not included in the `LS` output. See [“DlR subcommand—Obtain a list of directory entries” on page 159](#) to list entries for all types of data sets or files.

Context

- See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about pattern matching and about specifying data sets and files.
- To make a file group the current working directory, see [“CD subcommand—Change the directory on the remote host” on page 149](#).
- To a list of complete directory entries with auxiliary information about the files, see [“DlR subcommand—Obtain a list of directory entries” on page 159](#).

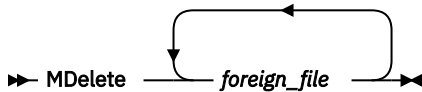
- To change the local directory, see “[LCd subcommand—Change the local working directory](#)” on page 175.

MDelete subcommand—Delete multiple files

Purpose

Use the MDelete subcommand to delete multiple files.

Format



Parameters

foreign_file

Specifies the name of the file to be deleted on the remote host.

Because more than one file can be deleted with the MDelete subcommand, the *foreign_file* parameter of the MDelete subcommand can be repeated many times, with each *foreign_file* separated by a blank space.

Results:

- If the LISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the server FTP.DATA file, the default is as if LISTSUBdir option was specified on the SITE subcommand.
- If the z/OS FTP server has the NOLISTSUBdir option on the SITE subcommand or LISTSUBDIR FALSE in the server FTP.DATA file, an MDelete * deletes only the files in the current directory.

Restriction: The LISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.

Examples

Example 1: Following is a sample entry and the response that displays after using the MDelete subcommand for multiple z/OS UNIX files.

```

cd '/u/user121/ftp.example'

>>>CWD '/u/user121/ftp.example'
250 HFS directory /u/user121/ftp.example is the current working directory
Command:
mdelete file1 file2 file3
>>>PORT 9,67,112,25,4,75
200 Port request OK.
>>>NLST file1
125 List started OK
250 List completed successfully.
>>>PORT 9,67,112,25,4,77
200 Port request OK.
>>>NLST file2
125 List started OK
250 List completed successfully.
>>>PORT 9,67,112,25,4,76
200 Port request OK.
>>>NLST file3
125 List started OK
250 List completed successfully.
>>>DELE file1
250 /u/user121/ftp.example/file1 deleted.
>>>DELE file2
250 /u/user121/ftp.example/file2 deleted.
>>>DELE file3
250 /u/user121/ftp.example/file3 deleted.
Command:

```

Example 2: MDelete * with SITE LISTSUBDir

Following is an example of MDelete * with the SITE LISTSUBDir option. This setting affects processing of the NLST command. The z/OS FTP client sends an NLST command to the server as part of MDelete * subcommand processing. The LISTSUBDir option specifies that both the current and the next subdirectory should be deleted as a result of processing an MDelete * subcommand. In this example, the current directory has a file x and a subdirectory y and subdirectory y has a file x.

```

site listsubdir
>>> SITE listsubdir
200 SITE command was accepted
prompt
Interactive mode is off
Command:
mdelete *
>>> PORT 127,0,0,1,4,15
200 Port request OK.
>>> NLST *
125 List started OK
250 List completed successfully.
>>> DELE x
250 /tmp/mgetmpu/x deleted.
>>> DELE y/x
250 /tmp/mgetmpu/y/x deleted.
Command:

```

Example 3: MDelete * with SITE NOLISTSUBDir

Following is an example of MDelete * with the SITE NOLISTSUBDir option. This setting affects processing of the NLST command. The z/OS FTP client sends an NLST command to the server as part of MDelete * subcommand processing. The NOLISTSUBDir option specifies that only the current directory should be deleted as a result of processing an MDelete * subcommand. In this example, the current directory has a file x and a subdirectory y and subdirectory y has a file x.

```

site Nolistsubdir
>>> SITE Nolistsubdir
200 SITE command was accepted
prompt
Interactive mode is off
Command:
mdelete *
>>> PORT 127,0,0,1,4,15
200 Port request OK.
>>> NLST *
125 List started OK
250 List completed successfully.
>>> DELE x
250 /tmp/mgetmput/x deleted.
Command:

```

Usage

- If you specify one or more incorrect foreign files with the MDelete subcommand, an error message specifying the incorrect foreign file is displayed. All correct foreign files are deleted, regardless of any incorrect foreign files, and the MDelete subcommand does not need to be reissued for these files.
- z/OS UNIX file names require special handling for certain special characters. All special characters that the operating system requires to be preceded by an escape character in commands issued to the shell must be preceded by the backslash (\) escape character, except for the single quote ('), double quote ("), or blank ().

Context

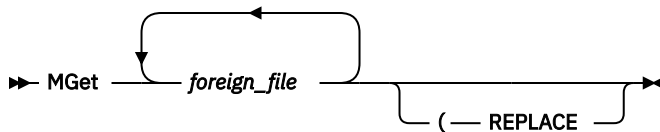
See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about naming conventions.

MGet subcommand—Copy multiple files

Purpose

Use the MGet subcommand to copy multiple files from a remote host to your local host and create a corresponding number of local files.

Format



Parameters

foreign_file

Specifies the name of the file to be retrieved from the remote host.

Because more than one file can be copied with the MGet subcommand, the *foreign_file* parameter of the MGet subcommand can be repeated many times, with each *foreign_file* separated by a blank space. You can use special characters for pattern matching when specifying the *foreign_file* with the MGet subcommand. These characters are dependent on the foreign host FTP server.

REPLACE

Causes a data set on your local host to be overwritten if it already exists. If the data set already exists, and you do not use the REPLACE parameter, the existing data set is not overwritten. A message informing you of this is displayed.

If the data set already exists and you specify the REPLACE parameter, the data in the file is overwritten, but the file is not reallocated; the local data set retains its existing characteristics.

Restriction: If you have configured UNIXFILETYPE=FIFO, the REPLACE parameter is not allowed.

Examples

The following is a sample entry and response that displays after using the MGet subcommand for multiple z/OS UNIX files.

```
cd '/u/user121/ftp.example'

>>>CWD '/u/user121/ftp.example'
250 HFS directory /u/user121/ftp.example is the current working directory
Command:
mget file1 file2 file3
>>>PORT 9,67,112,25,4,90
200 Port request OK.
>>>NLST file1
125 List started OK
250 List completed successfully.
>>>PORT 9,67,112,25,4,91
200 Port request OK.
>>>NLST file2
125 List started OK
250 List completed successfully.
>>>PORT 9,67,112,25,4,92
200 Port request OK.
>>>NLST file3
125 List started OK
250 List completed successfully.
>>>PORT 9,67,112,25,4,93
200 Port request OK.
>>>RETR file1
125 Sending data set /u/user121/ftp.example/file1
250 Transfer completed successfully.
3464 Bytes transferred in 1.031 seconds. Transfer rate 3.36 kbytes/sec.
>>>PORT 9,67,112,25,4,94
200 Port request OK.
>>>RETR file2
125 Sending data set /u/user121/ftp.example/file2
250 Transfer completed successfully.
3993 Bytes transferred in 0.923 seconds. Transfer rate 4.33 kbytes/sec.
>>>PORT 9,67,112,25,4,95
200 Port request OK.
>>>RETR file3
125 Sending data set /u/user121/ftp.example/file3
250 Transfer completed successfully.
3993 Bytes transferred in 0.791 seconds. Transfer rate 5.05 kbytes/sec.
Command:
```

The following is a sample entry and response that displays after using the MGet subcommand using a wildcard character in the file name.


```

Command:
mget file*
>>>PORT 9,67,113,57,5,123
200 Port request OK.
>>>NLST file*
125 List started OK
250 List completed successfully.
Mget file1 (Yes|No|Quit|Stop prompting)? s
>>>PORT 9,67,113,57,5,124
200 Port request OK.
>>>RETR file1
125 Sending data set /u/user31/file1
250 Transfer completed successfully.
164 bytes transferred in 0.310 seconds. Transfer rate 0.53 Kbytes/sec.
>>>PORT 9,67,113,57,5,125
200 Port request OK.
>>>RETR file2
125 Sending data set /u/user31/file2
250 Transfer completed successfully.
164 bytes transferred in 0.270 seconds. Transfer rate 0.61 Kbytes/sec.
>>>PORT 9,67,113,57,5,126
200 Port request OK.
>>>RETR file3
125 Sending data set /u/user31/file3
250 Transfer completed successfully.
164 bytes transferred in 0.280 seconds. Transfer rate 0.59 Kbytes/sec.
Command:

```

Results:

- When you use the MGet subcommand, FTP might truncate data records and you might lose data if:
 - You are creating a new data set at the client and the value of LRecl, as shown by the LOCStat command, is a value less than the LRecl of a received data set, FTP truncates the received data set.
 - The data set name already exists at the client and the logical record length (LRecl) of the data set at the client is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.

You can encounter this situation when you use MGet with the REPLACE option.
- If the name specified for *foreign_file* is not acceptable to your local host, the file is not transferred. To a file from the remote host, you must have a defined working directory on that host, and you must have read privileges to the files in this working directory.
- If you specify one or more incorrect foreign files with the MGet subcommand, an error message specifying the incorrect foreign file is displayed. All correct foreign files are retrieved, regardless of any incorrect foreign files, and do not need to be reissued.
- z/OS UNIX file names require special handling for certain special characters. Except for single quote ('), double quote ("), or blank (), all special characters that the operating system requires to be preceded by an escape character in commands issued to the shell must be preceded by the backslash (\) escape character.
- The MGet subcommand is not applicable to generation data groups (GDGs).
- The MGet subcommand can be used with the PROXY subcommand to transfer files from a host on a primary connection to a host on a secondary connection. See [“PROXY subcommand—Execute FTP subcommand on secondary control connections”](#) on page 235 for more information.
- If the data set is migrated, it is replaced regardless of the replace option.
- The MGet subcommand removes all directory information from remote file names. This causes all the files to be saved in the same z/OS UNIX file system directory when transferring into a z/OS UNIX file system. The directory structure of the remote host will not be preserved.
- If the LISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the server FTP.DATA file, the default is as if LISTSUBdir option was specified on the SITE subcommand.
- If the z/OS FTP server has the NOLISTSUBdir option specified on the SITE subcommand or has LISTSUBDIR FALSE specified in the server FTP.DATA file, an mget * command gets only the files in the current directory.

- The LISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.
- When UNIXFILETYPE=FIFO is configured and the local directory is a z/OS UNIX directory, the following apply:
 - New files are created as named pipes.
 - Transfers into existing z/OS UNIX regular files will fail.
 - Whether the named pipe is new or existing, FTP cannot write to the named pipe until another process on the z/OS client host opens the named pipe for reading. The z/OS FTP client waits up to the number of seconds specified by the FIFOOPEN TIME value for another process to open the named pipe.
 - FTP waits up to the number of seconds specified by the FIFOIOTIME value for each write to the named pipe to complete. The client does not block during writes unless it writes to the named pipe much faster than the named pipe reader reads from the pipe. If the client cannot write any data to the named pipe for the number of seconds specified by the FIFOIOTIME value, it fails the file transfer.
 - Data that is transferred into an existing named pipe is appended to the contents of the named pipe.

Guideline: When you transfer files into a z/OS UNIX directory, the configured UMASK value determines the new file permissions. Code the UMASK statement in the FTP.DATA file or issue the LOCSITE UMASK subcommand to configure the UMASK value.

Related topics:

- See [“CD subcommand—Change the directory on the remote host”](#) on page 149 for more information about working directories.
- See [Appendix A, “Specifying data sets and files,”](#) on page 359 for more information about naming conventions.
- See [“Using z/OS UNIX System Services named pipes”](#) on page 104 for more details about named pipes.

MKdir subcommand—Create a directory on the remote host

Purpose

Use the MKdir subcommand to create a PDS, PDSE, or z/OS UNIX directory on the remote host.

Format

```
➤ MKdir — directory ( — like — local_directory )
```

Parameters

directory

Specifies the name of the remote directory to be created.

local_directory

Specifies the name of a local directory that is to be a model for the remote directory.

Requirements:

- *local_directory* and *directory* must be MVS directories.
- The server must be a Communication Server for z/OS FTP server.

Guidelines:

- Use this parameter only when you have called the FTP client interactively.
- You can use this parameter to specify a local MVS directory that has characteristics you want for the remote directory. For example, use the (like parameter when allocating the target of a load module transfer to ensure that the source and target directories are compatible for load module transfer.

Restrictions:

- Only the 3390 device architecture is supported. Unpredictable results will occur if the source or target directory is on devices that use a different architecture.
- FTP can only approximate the following characteristics of *local_directory*: SPACETYPE, DIRECTORY, PRIMARY, and SECONDARY. Thus, the corresponding characteristics of *directory* might not match the original allocation of *local_directory*. For complete control over these characteristics, do not use the (like parameter.

Results:

- FTP must read *local_directory* to determine its characteristics. Do not use the (like parameter if this is not acceptable.
- FTP sends a SItE command to the server for you to configure the FTP server to allocate *directory* with the same characteristics as *local_directory*. A SItE command changes the server configuration for the rest of the session; consequently, the server configuration changes for the rest of the session when you specify the (like parameter. Do not use the (like parameter if you do not want the server configuration to change.
- If *local_directory* is a migrated data set, FTP checks the local AUTORECALL setting to determine whether to recall the data set or fail the request.
 - If AUTORECALL is true, FTP tries to recall the migrated data set.
 - If AUTORECALL is false, FTP fails the MKdir request.

You can change the local AUTORECALL setting with the LOCStE subcommand. Choosing AUTORECALL might result in a long delay when the FTP client waits for the data set to become available.

- If *local_directory* is a data set that is not mounted, FTP checks the local AUTOMOUNT setting to determine whether to mount the data set or fail the request.
 - If AUTOMOUNT is true, FTP tries to mount the data set.
 - If AUTOMOUNT is false, FTP fails the MKdir request.

You can change the local AUTOMOUNT setting with the LOCStE subcommand. Choosing AUTOMOUNT might result in a long delay when the client waits for the data set to become available.

Results

The MKdir subcommand directs the FTP client to send an MKD command to the remote host FTP server to create a directory with name *directory*.

- If the server is a Communications Server for z/OS FTP server, and *directory* is a fully qualified MVS data set name, the server allocates a PDS or PDSE named *directory*. For example, MKdir 'USER33.TEST.PDS' causes the server to create a PDS or PDSE named 'USER33.TEST.PDS'.
- If the server is a Communications Server for z/OS FTP server, and *directory* is an absolute path name, the server creates a z/OS UNIX directory named *directory*. For example, mkdir/tmp/logs directs the server to create a z/OS UNIX directory named /tmp/logs.

Otherwise, the current working directory at the remote host determines whether the FTP server interprets *directory* as an MVS low level qualifier (LLQ) or as a z/OS UNIX relative path name.

- If the current working directory is an MVS directory, the server allocates a PDS or PDSE in the current working directory whose LLQ is *directory*. For example, suppose the current working directory is 'USER33.TEST'. The subcommand MKdir PDS directs the server to create a PDS or PDSE called 'USER33.TEST.PDS'.
- If the current working directory is a z/OS UNIX directory, the server creates a subdirectory within the current working directory. For example, suppose the current working directory is /tmp. The subcommand MKdir logs direct the server to create the directory /tmp/logs.

For a z/OS FTP server, the characteristics of the z/OS UNIX directory or MVS directory are determined by the server's configuration. For example, when the server is allocating an MVS directory, the server PDSTYPE configuration option specifies whether the server allocates a PDS or PDSE. You can use the STatus subcommand to display the server configuration and the SItE subcommand to change the server configuration.

Examples

In this example, a directory is created on the remote host (1.1.2.3 in this example). Both EXAMPLE and FTP.EXAMPLE are created in the remote host, showing the difference between specifying and omitting quotation marks in the directory name.

```
User: ftp 9.67.113.24.621
System: IBM FTP CS V1R5
        FTP: using TCPCS
        FTP.DATA FILE NOT FOUND. USING HARDCODED DEFAULT VALUES.
        Connecting to 1.1.2.3, port 21
        220-EZAFTSRV IBM FTP CS V1R2 at EMU.ABC.OZ, 15:34:32 on 2000-08-03.
        220 Connection will not timeout.
        NAME (<host>:tsuserid):

User: user121
System: >>>USER user121
        331 Send password please.
        Password:

User: >>>PASS *****
        230 USER121 is logged on. Working directory is '/u/user121'.
        Command:
```

```
User: dir
System: >>>PORT 9,67,112,25,4,96
        200 Port request OK.
        >>>LIST
        125 List started OK.
        total 2768
        -rwxr-xr-t 2 USER121 SYS1 389120 Feb 5 16:03 ftpdka
        -rwxr-xr-t 2 USER121 SYS1 962560 Feb 5 16:04 ftpsrvka
        -rw-r----- 1 USER121 SYS1 11648 Jan 20 14:30 g.s
        drwxr-x--- 3 USER121 SYS1 0 Oct 21 17:50 msg
        -rw-r----- 1 USER121 SYS1 1458 Jan 10 19:25 s.k
        drwxr-x--- 2 USER121 SYS1 0 Feb 6 15:59 tcpip
        drwxr-x--- 2 USER121 SYS1 0 Feb 6 17:29 test
        250 List completed successfully.
        Command:

User: mkdir example
System: >>>MKD example
        257 "/u/user121/example" created.
        Command:
```

```
User: dir
System: >>>PORT 9,67,112,25,4,97
        200 Port request OK.
        >>>LIST
        125 List started OK.
        total 2768
        drwxr-xr-- 2 USER121 SYS1 0 Feb 7 19:57 example
        -rwxr-xr-t 2 USER121 SYS1 389120 Feb 5 16:03 ftpdka
        -rwxr-xr-t 2 USER121 SYS1 962560 Feb 5 16:04 ftpsrvka
        -rw-r----- 1 USER121 SYS1 11648 Jan 20 14:30 g.s
        drwxr-x--- 3 USER121 SYS1 0 Oct 21 17:50 msg
        -rw-r----- 1 USER121 SYS1 1458 Jan 10 19:25 s.k
        drwxr-x--- 2 USER121 SYS1 0 Feb 6 15:59 tcpip
        drwxr-x--- 2 USER121 SYS1 0 Feb 6 17:29 test
        250 List completed successfully.
        Command:

User: mkdir '/u/user121/ftp.example'
System: >>>MKD '/u/user121/ftp.example'
        257 "/u/user121/ftp.example" created.
        Command:
```

```

User:  dir
System: >>>PORT 9,67,112,25,4,98
        200 Port request OK.
        >>>LIST
        125 List started OK.
        total 2800
drwxr-x---  2 USER121 SYS1      0 Feb  7 19:57 example
drwxr-x---  2 USER121 SYS1      0 Feb  7 19:57 ftp.example
-rwxr-xr-t  2 USER121 SYS1 389120 Feb  5 16:03 ftpdka
-rwxr-xr-t  2 USER121 SYS1 962560 Feb  5 16:04 ftpsrvka
-rw-r----- 1 USER121 SYS1  11648 Jan 20 14:30 g.s
drwxr-x---  3 USER121 SYS1      0 Oct 21 17:50 msg
-rw-r----- 1 USER121 SYS1  1458 Jan 10 19:25 s.k
drwxr-x---  2 USER121 SYS1      0 Feb  6 15:59 tcpip
drwxr-x---  2 USER121 SYS1      0 Feb  6 17:29 test
        250 List completed successfully.
Command:

```

Related topics:

- See [“Dynamic allocation of new data sets”](#) on page 82 for information about FTP configuration options that determine data set characteristics when creating an MVS directory.
- See [“SItE subcommand—Send site-specific information to a host ”](#) on page 247 for information about setting the SItE configuration options.
- See [“STaTus subcommand—Retrieve status information from a remote host”](#) on page 278 for information about displaying the SItE configuration options.
- See [“SENDSite subcommand—Toggle the sending of site information ”](#) on page 246 for information about SENDSite subcommand.
- See [“LOCSItE subcommand—Specify site information to the local host”](#) on page 179 for information about changing the AUTORECALL and AUTOMOUNT settings.

MKFifo subcommand—Create a named pipe at the FTP server host

Purpose

Use the MKFifo subcommand to create a z/OS UNIX named pipe on the remote host.

Requirement: The FTP server on the remote host must be z/OS V1R11 or later.

Format

➤ MKFifo — *pathname* ➤

Parameters

pathname

Specifies the path name of the z/OS UNIX named pipe that is to be created. You can specify a relative path name or an absolute path name.

Examples

In this example, the named pipe, /tmp/named.pipe, is created by specifying an absolute path name:

```

Command:
cd 'USER1'
>>> CWD 'USER1'
250 "USER1." is the working directory name prefix.
Command:
mkfifo /tmp/named.pipe
>>> XFIF /tmp/named.pipe
257 named pipe /tmp/named.pipe created
Command:

```

In this example, the named pipe, my named pipe, is created by specifying a relative path name:

```
cd /tmp
>>> CWD /tmp
250 HFS directory /tmp is the current working directory.
Command:
mkfifo my named pipe
>>> XFIF my named pipe
257 named pipe /tmp/my named pipe created
Command:
```

This directory listing shows the named pipes that are created by these examples.

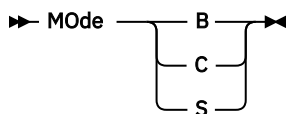
```
Command:
dir /tmp/*pipe
>>> PORT 9,42,105,36,4,29
200 Port request OK.
>>> LIST /tmp/*pipe
125 List started OK
prwxr-x--- 1 OMVSKERN OMVSGRP      0 Jun 10 02:28 /tmp/my named pipe
prwxr-x--- 1 OMVSKERN OMVSGRP      0 Jun 10 02:33 /tmp/named.pipe
250 List completed successfully.
Command:
```

MMode subcommand—Set the data transfer mode

Purpose

Use the MMode subcommand to define how bits of data are to be transmitted.

Format



Parameters

B

Sets the block mode. In block mode, data is transmitted as a series of data blocks, preceded by one or more header bytes. Block mode preserves the logical record boundaries of the data set or file. When MMode is set to B, the data transfer type must be EBCDIC.

Specifying MMode B is equivalent to specifying the BLock subcommand.

C

Sets the compressed mode. In compressed mode, data is transmitted as a series of data blocks, preceded by one or more header bytes. Compressed mode preserves the logical record boundaries of the data set or file. In compressed mode, data is transmitted without repetitive characters and blanks. When MMode is set to C, the data transfer type must be EBCDIC.

Note: Because additional processing time is required for both the sender and receiver to compress or decompress the data, evaluate the time factor before you compress a file.

Specifying MMode C is equivalent to specifying the COMpress subcommand.

S

Sets the stream mode. In stream mode, data is transmitted as a stream of bytes. Any data transfer type can be used with stream mode. Stream mode is efficient because data block information is not transferred.

Specifying MMode S is equivalent to specifying the STREam subcommand.

Usage

- To use MMode C, the receiving host must support the compressed data mode.
- Data compression increases CPU processing costs even if the amount of data transferred is not large.

Context

- For the syntax of the BLock, COMpress, and STREam subcommands, see [“BBlock subcommand—Set the block data transfer mode”](#) on page 148, [“COMpress subcommand—Set the compressed data transfer mode”](#) on page 153, or [“STREam subcommand—Set the stream data transfer mode”](#) on page 285.
- For more information about transfer methods, see [Table 13 on page 46](#).

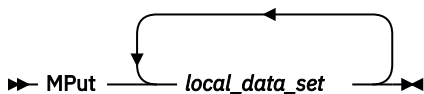
Tip: Use MODE B or MODE C in conjunction with a CHKConfidence value of TRUE in FTP.DATA to improve detection of incomplete file transfers.

MPut subcommand—Copy multiple data sets to the remote host

Purpose

Use the MPut subcommand to copy multiple data sets from your local host to the remote host.

Format



Parameters

local_data_set

Specifies the name of the file on your local host being sent to the remote host.

Because more than one data set can be copied with the MPut subcommand, the *local_data_set* parameter of this subcommand can be repeated many times, with each *local_data_set* separated by a blank space. You can use the asterisk (*) character for pattern matching when specifying the *local_data_set* with the MPut subcommand.

When the wildcard symbol (*) is used in the filename parameter, and the GLOB subcommand is set to expand metacharacters in file names, the LISTSUBdir option affects the result of MPut. For more information about the LISTSUBdir option, see [“LOCSite subcommand—Specify site information to the local host”](#) on page 179, or the [LISTSUBDIR statement \(FTP client and server\)](#) details in [z/OS Communications Server: IP Configuration Reference](#).

Examples

The following is a sample entry and response that displays after using the MPut subcommand to send selected files.

```

Command:
mput file*
Mput FILE1 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,128
200 Port request OK.
>>>STOR FILE1
125 Storing data set /u/user31/temp/FILE1
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds. Transfer rate 16.40 Kbytes/sec.
Mput FILE2 (Yes|No|Quit|Stop prompting)? no
Mput FILE3 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,129
200 Port request OK.
>>>STOR FILE3
125 Storing data set /u/user31/temp/FILE3
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds. Transfer rate 16.40 Kbytes/sec.
Command:
dir
>>>PORT 9,67,113,57,5,130
200 Port request OK.
>>>LIST
125 List started OK
total 16
-rw-r----- 1 USER31 SYS1      162 Aug 14 13:20 FILE1
-rw-r----- 1 USER31 SYS1      162 Aug 14 13:21 FILE3
250 List completed successfully.
Command:

```

Results:

- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the MPut subcommand, FTP might truncate data records and you might lose data, if:
 - You are creating a new file at the server and the value of LRecl, as shown by the STATUS subcommand, is a value less than the LRecl of the transmitted data set and SENDSITE has been set to OFF, FTP truncates the transmitted data set.
 - The data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.
- By default, if you use the MPut subcommand, the remote host creates files with the same names as those specified in *local_data_set* and overwrites any existing files with those names.

To put files on the remote host with unique file names, you must have set unique storage on before issuing the MPut command. Use the SUnique subcommand to change the storage method.
- If you specify one or more incorrect parameters with the MPut subcommand, an error message specifying the incorrect parameter is displayed. All correct files are transferred, regardless of any incorrect parameters, and do not need to be reissued.
- When UNIXFILETYPE=FIFO is configured at the FTP client, and the local directory is a z/OS UNIX directory, the following apply:
 - Named pipes are transferred; transfers from existing z/OS UNIX regular files will fail.
 - FTP is unable to read from the named pipe until another process on the z/OS client host opens the named pipe for writing. The z/OS FTP client waits up to the number of seconds specified by the FIFOOPEN TIME value for another process to open the named pipe.
 - FTP waits up to the number of seconds specified by the FIFOIO TIME value for each read from the named pipe to complete. If the client cannot read any data from the named pipe for the number of seconds specified by the FIFOIO TIME value, the file transfer fails.
 - Sending the named pipe permanently removes data from the named pipe.
- If the FTP server is a z/OS FTP server, the server UNIXFILETYPE configuration option is set to FIFO, and the remote file directory is a z/OS UNIX file system directory, the server creates the remote files as named pipes rather than as regular files.

The FTP server creates named pipes using the same names as those specified in the local file and appends to existing named pipes with those names. The FTP server rejects transfers into z/OS UNIX regular files with the same names as those specified in the local file.

For more information about using z/OS UNIX named pipes, see [“Using z/OS UNIX System Services named pipes”](#) on page 104.

Requirement: To send a data set to the remote host, you must have a defined working directory on the remote host and write privileges to the files in this working directory.

Restriction: The MPut subcommand is not applicable to generation data groups (GDGs).

Related topics:

- See [“SUnique subcommand—Changes the storage method ”](#) on page 286 for information about changing the storage method on the remote host.
- See [Appendix A, “Specifying data sets and files,”](#) on page 359 for more information about naming conventions.
- MPut can be used with the PROXy subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See [“PROXy subcommand—Execute FTP subcommand on secondary control connections”](#) on page 235 for more information.

MVSGet subcommand – Copy a remote data set into a local data set with the remote data set attributes

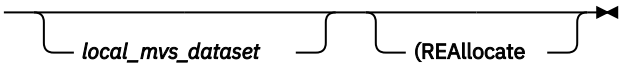
Purpose

Use the MVSGet subcommand to transfer MVS data sets from a z/OS FTP server to a z/OS FTP client without knowing the details of the server data set allocation. The MVS data set can be one of the following data set types:

- z/OS physical sequential data set
- z/OS PDS or library
- z/OS generation data set reference

For a physical sequential data set, MVSGet works like a combination of the LOCSite and Get subcommands. For a partitioned data set, MVSGet works like a combination of the LOCSite, LMkir (like <remote directory>, and MGet * subcommands. All the data set members are transferred. Regardless of the type of data set that is transferred, FTP reconfigures the client to allocate the local data set with the same attributes as the remote data set.

Format

➡ MVSGet — *remote_mvs_dataset* 

Parameters

remote_mvs_dataset

Specifies the name of the data set to be retrieved from the remote MVS host. If the server working directory is a data set prefix, the remote file is a data set with the *remote_mvs_dataset* name appended to the server working directory. You can override the server working directory in the remote file name by specifying the *remote_mvs_dataset* value as a complete data set name that is enclosed in single quotation marks.

local_mvs_dataset

Specifies the name of the local MVS data set that is created as a result of the MVSGet subcommand. If the local working directory is a data set prefix, the local file is a data set with the *local_mvs_dataset* name appended to the local working directory. You can override the local working directory in the

local file name by specifying the *local_mvs_dataset* value as a complete data set name that is enclosed in single quotation marks. If the *local_mvs_dataset* parameter is not specified, the value is the same as the *remote_mvs_dataset* value.

REALlocate

Causes the MVS data set on your local MVS host to be deleted and reallocated with the attributes of the remote MVS data set, if the local MVS data set is an existing data set. If the MVS data set already exists and you do not use the REALlocate parameter, the existing data set is not deleted and reallocated, the MVSGet subcommand fails, and a message is displayed.

The following example shows using the MVSGet subcommand to retrieve a PDS data set.

```
mvsget 'user1.remote.pds' 'user1.local.pds' (REALlocate
EZA1701I >>> XDSS 'user1.remote.pds'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMARY=1 SECO
NDARY=1 TRACKS EATTR=SYSTEM
EZZ9815I local site variables have changed
EZA2245I "USER1.LOCAL.PDS" created.
EZA2081I Local directory name set to partitioned data set USER1.LOCAL.PDS
EZA1701I >>> PWD
257 "'USER1.'" is working directory.
EZA1701I >>> CWD 'user1.remote.pds'
250 The working directory "USER1.REMOTE.PDS" is a partitioned data set
EZA1701I >>> PORT 127,0,0,1,4,5
200 Port request OK.
EZA1701I >>> NLST *
125 List started OK.
250 List completed successfully.
EZA1701I >>> PORT 127,0,0,1,4,6
200 Port request OK.
EZA1701I >>> RETR NEW1
125 Sending data set USER1.REMOTE.PDS(NEW1)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.010 seconds.
Transfer rate 13.40 Kbytes/sec.
EZA1701I >>> PORT 127,0,0,1,4,7
200 Port request OK.
EZA1701I >>> RETR NEW2
125 Sending data set USER1.REMOTE.PDS(NEW2)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.010 seconds.
Transfer rate 13.40 Kbytes/
sec.
EZA2581I Local HFS directory is /u/user1.
EZA1701I >>> CWD 'USER1.'
250 "USER1." is working directory name prefix
EZA2108I Confidence=High for MVSGET of USER1.LOCAL.PDS
```

The following example shows using the proxy MVSGet subcommand to transfer a library between two servers.

```
proxy mvsget 'user.linklib' 'user1.local.pdse' (REALlocate
EZA1701I >>> PWD
257 "'USER2.'" is working directory.

EZA1701I >>> XDSS 'user.linklib'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDSE RECFM=U BLKSIZE=32760 DIRECTORY=3 LRECL=256 PRIMARY=20 SEC
ONDARY=1 CYLINDERS EATTR=SYSTEM
EZA1701I >>> XDSS 'user1.local.pdse'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDSE RECFM=U BLKSIZE=32760 DIRECTORY=3 LRECL=256 PRIMARY=20 SEC
ONDARY=1 CYLINDERS EATTR=SYSTEM
EZA1701I >>> DELE 'user1.local.pdse'
250 USER1.LOCAL.PDSE deleted.
EZA1701I >>> SITE PDSTYPE=PDSE RECFM=U BLKSIZE=32760 DIRECTORY=3 LRECL=256 PRIMA
RY=20 SECONDARY=1 CYLINDERS EATTR=SYSTEM
200 SITE command was accepted
EZA1701I >>> MKD 'user1.local.pdse'
257 "'USER1.LOCAL.PDSE'" created.
EZA1701I >>> CWD 'user1.local.pdse'
250-The working directory may be a load library
250 The working directory "USER1.LOCAL.PDSE" is a partitioned data set
EZA1701I >>> PWD
257 "'USER1.'" is working directory.
EZA1701I >>> CWD 'user.linklib'
```

```

250-The working directory may be a load library
250 The working directory "USER.LINKLIB" is a partitioned data set
EZA1701I >>> XLMT QUERY 0 *
250 PDSE 12787712 - send next command for load module transfer
EZA1701I >>> XLMT PDSE 12787712
250 PDSE 12787712 - send next command for load module transfer
EZA1701I >>> PASV
227 Entering Passive Mode (127,0,0,1,4,19)
EZA1701I >>> PORT 127,0,0,1,4,19
200 Port request OK.
EZA1701I >>> RETR load module
125-Transferring load module
125 DCB 32768 32760
EZA1701I >>> XLMT DCB 32768 32760

250 DCB saved, send next command for load module transfer
EZA1701I >>> STOR load module
125 Transferring load module
250 Transfer completed successfully.
250 Transfer completed successfully.
EZA1701I >>> CWD 'USER2.'
250 "USER1." is working directory name prefix
EZA1701I >>> CWD 'USER1.'
250 "USER1." is working directory name prefix

```

Restrictions:

- The MVSGet subcommand supports only these data set types.
 - z/OS physical sequential data set
 - z/OS PDS or library data set
 - z/OS generation data set reference
- The MVSGet subcommand does not support transfer of an empty PDS or library. You can use the LMkdir subcommand with the (like parameter for that purpose.
- The MVSGet subcommand supports checkpointing for block mode restart of an interrupted file transfer. However, if you transfer a PDS or library data set, the target data set is deleted if the transfer fails. You cannot use the REStart subcommand to restart these transfers.
- The target generation data set must be a positive reference and cannot be a library data set.
- If the source data set is a PDS, the target generation data set must be referenced with its absolute name.
- The MVSGet subcommand does not support specifying the local data set as a ddname.
- If the source data set is in physical sequential extended format, the target data set is allocated as if the DSNTYPE parameter with SYSTEM value was configured. If the system default DSNTYPE value is not EXTREQ or EXTPREF, the source data set might exceed the architecture size limitation of the system default DSNTYPE value and the transfer fails.
- FTP can determine only approximate values for the primary allocation, secondary allocation, and space type, but it uses an allocation that is large enough to contain the data. FTP primary space calculation behaves similar to how z/OS allocates a data set when you specify the LIKE parameter on a JCL DD statement. The primary space to be allocated for the new data set is determined by adding up the space allocated in the first three extents on the initial volume of the source data set. See *LIKE parameter* in [z/OS MVS JCL Reference](#) for more information about the LIKE parameter. For complete control over the initial allocation, use the LOCStc subcommand with the Get subcommand instead of the MVSGet subcommand.
- If the target tape data set does not exist on the tape volume, the transfer sometimes succeeds with the MVSGet subcommand. However, the MVSGet subcommand does not support reallocating an existing tape data set.
- For PDS and library data sets, FTP must read the directory of the source data set at least twice when using the MVSGet subcommand.

Results:

- If the target local data set already exists on the FTP client without REALLOCATE specified, the MVSGet subcommand fails.

- FTP ignores the GLOB subcommand toggle when using the MVSGET subcommand. MVSGET works as if the GLOB subcommand is always toggled on.
- The MVSGET subcommand of a PDS or library data set gets the data set as a whole data set and gets all the members of it to the local PDS or library data set. The MVSGET subcommand is not prompted before transferring each member, regardless of whether the PROMPT subcommand toggle is set to interactive mode.
- If an FTP file transfer ends prematurely for a physical sequential data set, the new data set that is created on the local host is disposed according to the CONDDISP configuration on the local host. See [“LOCSITE subcommand—Specify site information to the local host” on page 179 or CONDDISP \(FTP client and server\) statement in z/OS Communications Server: IP Configuration Reference](#) for more information about the CONDDISP configuration option. However, if you transfer a PDS or library data set, the new data set that is created on the local host is deleted regardless of the CONDDISP configuration if the transfer ends prematurely.
- The MVSGET subcommand changes the FTP client configuration so that the subcommand can allocate the local data set as the remote data set.
- The MVSGET subcommand can determine only an approximate size of the source data set when allocating the target data set, but the target data set is large enough to complete the transfer. For complete control over the initial allocation, use the LOCSITE subcommand with the GET subcommand.
- If the remote source data set is migrated, the server inspects the AUTORECALL setting to decide whether to recall the data set or to fail the request. If AUTORECALL is set to true, FTP attempts to recall the data set; otherwise, it fails the request. Similarly, if the remote data set is not mounted, the server inspects the AUTOMOUNT setting to decide whether to mount the data set or to fail the request. If AUTOMOUNT is set to true, the server attempts to mount the data set; otherwise, it fails the request. You can change the AUTOMOUNT and AUTORECALL settings of the server with the SITE subcommand. Choosing AUTOMOUNT or AUTORECALL can cause a long delay because the server waits for the data set to become available.

Requirements:

- The local and remote data sets must be MVS data sets.
- The remote FTP server must be z/OS V2R1 Communications Server or later releases.
- Users must have READ access to the source data set and ALTER access to the target data set to use the MVSGET subcommand.

Guidelines:

- An FTP client can specify one or more of the Storage Management Subsystem (SMS) classes to manage characteristics that are associated with or assigned to data sets. See [“Specifying values for new data sets - Storage Management System\(SMS\)” in z/OS Communications Server: IP Configuration Guide](#) for more information about SMS classes.
- The MVSGET subcommand can determine only an approximate size of the source data set when allocating the target data set. For complete control over the initial allocation, use the LOCSITE subcommand with the GET subcommand.
- The MVSGET subcommand changes the FTP client configuration as if the user issued the LOCSITE and LCd subcommands. Restart the FTP client to reinstate the initial configuration or use the LOCSITE and LCd subcommands to configure the client.

Related topics:

- See [Appendix A, “Specifying data sets and files,” on page 359](#) for more information about naming conventions.
- You can use the MVSGET subcommand with the PROXY subcommand to transfer files from a host on a primary connection to a host on a secondary connection. See [“PROXY subcommand—Execute FTP subcommand on secondary control connections” on page 235](#) for more information.
- See [“LOCSITE subcommand—Specify site information to the local host” on page 179 or FTP configuration statements in FTP.DATA in z/OS Communications Server: IP Configuration Reference](#)

for more information about the BLKSIZE, DIRECTORY, DSNTYPE, EATTR, LRECL, PDSTYPE, PRIMARY, RECFM, SECONDARY and SPACETYPE configuration options.

MVSPut subcommand – Copy a local data set into a remote data set name with the local data set attributes

Purpose

Use the MVSPut subcommand to transfer MVS data sets from a z/OS FTP client to a z/OS FTP server without knowing the details of the client data set allocation. The MVS data set can be one of the following data set types:

- z/OS physical sequential data set
- z/OS PDS or library data set
- z/OS generation data set reference

For a physical sequential data set, MVSPut works like a combination of the Site and PUt subcommands. For a partitioned data set, MVSPut works like a combination of the Site, MKDir (like <local directory>, and MPut *. subcommands. Regardless of the type of data set that is transferred, FTP reconfigures the server to allocate the remote data set with the same attributes as the local data set.

Format

➔ MVSPut — *local_mvs_dataset* ————— *remote_mvs_dataset* (REALlocate)

Parameters

local_mvs_dataset

Specifies the name of the data set to be sent from the local MVS host to the remote MVS host. If the server working directory is a data set prefix, the local file is a data set with the *local_mvs_dataset* name appended to the local working directory. You can override the local working directory in the local file name by specifying the *local_mvs_dataset* value as a complete data set name that is enclosed in single quotation marks.

remote_mvs_dataset

Specifies the name of the remote MVS data set that is created by using the MVSPut subcommand. If the current server working directory is a data set prefix, the remote file is a data set with the *remote_mvs_dataset* name appended to the current server working directory. You can override the server working directory in the remote file name by specifying the *remote_mvs_dataset* value as a complete data set name enclosed in single quotation marks. If the *remote_mvs_dataset* parameter is not specified, the name of the remote MVS data set is the same as the *local_mvs_dataset* value.

REALlocate

Causes the MVS data set on your remote MVS host to be deleted and reallocated with the attributes of the local MVS data set, if the value is an existing MVS data set. If the MVS data set already exists and you do not use the REALlocate parameter, the existing data set is not deleted and reallocated, the MVSPut subcommand fails, and a message is displayed.

The following example shows using the MVSPut subcommand with the REALlocate parameter specified to transfer a physical sequential data set to the server.

```
mvspu 'user1.ps.source' 'user1.ps.target' (REALlocate
EZA1701I >>> XDSS 'user1.ps.target'
200- LASTREF=2011/12/07 DSEMPY=FALSE
200 SITE DSNTYPE=BASIC RECFM=VB BLKSIZE=6233 LRECL=256 PRIMARY=1 SECONDARY=1 TRA
CKS EATTR=SYSTEM
EZA1701I >>> DELE 'user1.ps.target'
250 USER1.PS.TARGET deleted.
EZA1701I >>> SITE DSNTYPE=BASIC RECFM=VB BLKSIZE=6233 LRECL=256 PRIMARY=1 SECON
DARY=1 TRACKS EATTR=SYSTEM
```

```

200 SITE command was accepted
EZA1701I >>> PORT 127,0,0,1,4,4
200 Port request OK.
EZA1701I >>> STOR 'user1.ps.target'
125 Storing data set USER1.PS.TARGET
250 Transfer completed successfully.
EZA1617I 2331 bytes transferred in 0.005 seconds.
Transfer rate 466.20 Kbytes/sec.

```

The following example shows a sample entry and response that is displayed after the MVSPut subcommand is used to transfer a PDS.

```

mvspu 'user1.local.pds' 'user1.remote.pds' (REAllocate
EZA1701I >>> PWD
257 "USER1." is working directory.
EZA1701I >>> XDSS 'user1.remote.pds'
200-LASTREF=2011/12/16 DSEMPY=FALSE
200 SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMARY=1 SECO
NDARY=1 TRACKS EATTR=SYSTEM
EZA1701I >>> DELE 'user1.remote.pds'
250 USER1.REMOTE.PDS deleted.
EZA1701I >>> SITE PDSTYPE=PDS RECFM=VB BLKSIZE=6233 DIRECTORY=27 LRECL=256 PRIMA
RY=1 SECONDARY=1 TRACKS EATTR=SYSTEM
200 SITE command was accepted
EZA2081I Local directory name set to partitioned data set USER1.LOCAL.PDS
EZA1701I >>> MKD 'user1.remote.pds'
257 "USER1.REMOTE.PDS" created.
EZA1701I >>> CWD 'user1.remote.pds'
250 The working directory "USER1.REMOTE.PDS" is a partitioned data set
EZA1701I >>> PORT 127,0,0,1,4,11
200 Port request OK.
EZA1701I >>> STOR NEW1
125 Storing data set USER1.REMOTE.PDS(NEW1)
250 Transfer completed
successfully.
EZA1617I 134 bytes transferred in 0.005 seconds.
Transfer rate 26.80 Kbytes/
sec.
EZA1701I >>> PORT 127,0,0,1,4,12
200 Port request OK.
EZA1701I >>> STOR NEW2
125 Storing data set USER1.REMOTE.PDS(NEW2)
250 Transfer completed successfully.
EZA1617I 134 bytes transferred in 0.005 seconds.
Transfer rate 26.80 Kbytes/
sec.
EZA2581I Local HFS directory is /u/user1.
EZA1701I >>> CWD 'USER1.'
250 "USER1." is working directory name prefix
EZA2108I Confidence=High for MVSPUT of USER1.LOCAL.PDS

```

Restrictions:

- The MVSPut subcommand supports only the following data set types:
 - z/OS physical sequential data set
 - z/OS PDS or library data set
 - z/OS generation data set reference
- The MVSPut subcommand does not support transfer of an empty PDS or library. You can use the MKDir subcommand with the (like parameter for that purpose.
- The MVSPut subcommand supports checkpointing for block mode restart of an interrupted file transfer. However, if you transfer a PDS or library data set, the target data set is deleted when the transfer fails. You cannot use the REStart subcommand to restart these transfers.
- The target generation data set must be a positive reference and cannot be a library data set.
- If the source data set is a PDS, the target generation data set must be referenced with its absolute name.
- If the source data set is in physical sequential extended format, the target data set is allocated as if the DSNTYPE parameter with SYSTEM value was configured. If the system default DSNTYPE value is not EXTREQ or EXTPREF, the source data set might exceed the architecture size limitation of the system default DSNTYPE value and the transfer fails.

- FTP can determine only approximate values for the primary allocation, secondary allocation, and space type, but it uses an allocation that is large enough to contain the data. FTP primary space calculation behaves similar to how z/OS allocates a data set when you specify the LIKE parameter on a JCL DD statement. The primary space to be allocated for the new data set is determined by adding up the space allocated in the first three extents on the initial volume of the source data set. See *LIKE parameter* in *z/OS MVS JCL Reference* for more information about the LIKE parameter. For complete control over the initial allocation, use the SItE subcommand with the PUt subcommand instead of the MVSPut subcommand.
- If the target tape data set does not exist on the tape volume, the transfer sometimes succeeds with the MVSPut subcommand. However, the MVSPut subcommand does not support reallocating an existing tape data set.
- For PDS and library data sets, FTP must read the directory of the source data set at least twice when using the MVSGet subcommand.

Results:

- FTP ignores the SendSite subcommand toggle when using the MVSPut subcommand. A SITE command is always triggered and sent to the FTP server.
- FTP ignores the SUnique subcommand setting when using the MVSPut subcommand. If the target remote data set exists on the FTP server without the REAllocate parameter specified, the MVSPut subcommand fails no matter SUnique is turned on or off.
- FTP ignores the GLob subcommand toggle when using the MVSPut subcommand. MVSPut works as if the GLob command is always toggled on.
- The MVSPut subcommand of a PDS or library data set sends the data set as a whole data set and transfers all the members of it to the remote PDS or library data set.
- If an FTP file transfer ends prematurely for a physical sequential data set, the new data set that is created on the remote host is disposed according to the CONDDISP configuration on the remote host. See *“Site subcommand—Send site-specific information to a host”* on page 247 or CONDDISP (FTP client and server) statement in *z/OS Communications Server: IP Configuration Reference* for more information about the CONDDISP configuration option. However, if you transfer a PDS or library data set, the new data set that is created on the remote host is deleted regardless of the CONDDISP configuration if the transfer ends prematurely.
- The MVSPut subcommand changes the FTP server configuration so that the subcommand can allocate the remote data set as the local data set.
- The MVSPut subcommand can determine only an approximate size of the source data set when allocating the target data set, but the target data set is large enough to complete the transfer. For complete control over the initial allocation, use the SItE subcommand with the PUt subcommand.
- If the local source data set is migrated, the FTP client inspects the AUTORECALL setting to determine whether to recall the data set or to fail the request. If AUTORECALL is set to true, FTP attempts to recall the data set; otherwise, it fails the request. Similarly, if the local source data set is not mounted, the FTP client inspects the AUTOMOUNT setting to determine whether to mount the data set or to fail the request. If AUTOMOUNT is set to true, the FTP client attempts to mount the data set; otherwise, it fails the request. You can change the AUTOMOUNT and AUTORECALL settings of the FTP client with the LOCSItE subcommand. Choosing AUTOMOUNT or AUTORECALL could can a long delay because the FTP client waits for the data set to become available.

Requirements:

- The local and remote data sets must be MVS data sets.
- The remote FTP server must be z/OS V2R1 Communications Server or later releases.
- Users must have READ access to the source data set and ALTER access to the target data set to use the MVSPut subcommand.

Guidelines:

- An FTP server can specify one or more of the Storage Management Subsystem (SMS) classes to manage characteristics that are associated with or assigned to data sets. See *“Specifying values for new data*

sets - Storage Management System(SMS)” in [z/OS Communications Server: IP Configuration Guide](#) for more information about SMS classes.

- The MVSPut subcommand can determine only an approximate size of the source data set when allocating the target data set. For complete control over the initial allocation, use the SItE subcommand with the PUt subcommand instead of the MVSPut subcommand.

Related topics:

- See [Appendix A, “Specifying data sets and files,”](#) on page 359 for more information about naming conventions.
- You can use the MVSPut subcommand with the PROXy subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See [“PROXy subcommand—Execute FTP subcommand on secondary control connections”](#) on page 235 for more information.
- See [“SItE subcommand—Send site-specific information to a host ”](#) on page 247 or [FTP configuration statements in FTP.DATA](#) in [z/OS Communications Server: IP Configuration Reference](#) for more information about the BLKSIZE, DIRECTORY, DSNTYPE, EATTR, LRECL, PDSTYPE, PRIMARY, RECFM, SECONDARY and SPACETYPE configuration options.

NOop subcommand—Test the connection

Purpose

Use the NOop subcommand to determine whether the foreign host is still responding.

Format

➡ NOop ➡

Parameters

There are no parameters for this subcommand.

Examples

- If the foreign host is responding, you receive one of the following responses:
200 OK or 200 NOOP command successful.
- If the foreign host does not respond or is not connected, you receive appropriate error messages, such as:

```
EZA1534I Control connection with ipaddr dies.  
EZA1457I You must first issue the 'OPEN' command.
```

Usage

You can use the NOop command to keep a connection alive that would otherwise be disconnected if it were idle for longer than the system timeout period.

Open subcommand—Connect to the FTP server

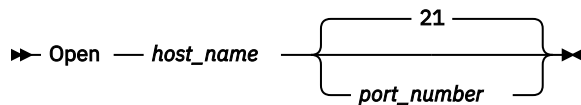
Purpose

Use the Open subcommand to open a connection to the remote host FTP server in the following situations:

- If, after closing a connection, you want to open another connection without leaving the FTP environment.

- If you were unable to open a connection when you specified a *foreign_host* value with the FTP command.

Format



Parameters

host_name

Specifies the host name or IP address of the foreign host. When using IPv6 link-local addresses, you can provide scope information along with the host name or IP address, as described in the [support for scope information](#) in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

port_number

Identifies a port on the foreign host. The default is well-known port 21.

Usage

If you are already connected to a host, you must disconnect from the host before you can connect to a different host with the Open subcommand. The only exception to this is if you are using the PROXY Open command. See “PROXY subcommand—Execute FTP subcommand on secondary control connections” on [page 235](#) for more information.

Context

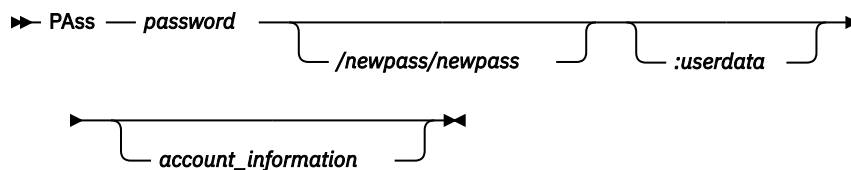
See “CLOSe subcommand—Disconnect from a remote host ” on [page 153](#) for more information about closing a connection.

PAss subcommand—Supply a password

Purpose

Use the PAss subcommand to supply a password or password phrase to a remote host.

Format



Parameters

password

Specifies your password or password phrase on the remote host used to log in to the FTP server.

/newpass/newpass

An optional parameter that resets a password or password phrase to *newpass*.

Requirement: If the security product of your FTP server host is RACF or another SAF-compliant security product, *password* and *newpass* must both be passwords, or both be password phrases.

:userdata

The optional user data must be separated from the password information by a colon (:). It can be any combination of up to 200 characters except the colon and the space (blank). Beware using the back slash character (\) in combination with other characters which might be interpreted as an escape sequence by the C compiler.

account_information

An optional parameter that will be supplied to the remote FTP server if the server requests account information after receiving *password*.

Result: Not all FTP servers support the *:userdata* parameter. The optional user data is used by the z/OS FTP server as a character string that is passed to the server's FTCHKPWD user exit routine. See [configuring the optional FTP user exits](#) in the *z/OS Communications Server: IP Configuration Guide* for more information.

Rules:

- The PAss subcommand must be preceded by the User subcommand. For some sites, the password completes your identification for access control on the remote host.
- Do not place any spaces between the passwords and the slashes (/), and the *:userdata* parameter.
- If the *password* or *newpass* parameter contains blanks, enclose the entire *password/newpass/newpass:userdata* sequence with quotation marks. If the password, newpass, or user data parameter itself contains a quotation mark, use the other style of quotation marks to enclose the parameters.

Examples:

pass "What's up, Doc?"

pass "What's up, Doc?/Not much; you?/Not much; you?"

pass "What's up, Doc?/Not much; you?/Not much; you?:I-coded-userdata-today"

but not:

pass 'What's up, Doc?'

pass "What's up, Doc?"/"Not much; you?"/"Not much; you?"

pass "What's up, Doc?"/Not-much;-you?/Not-much;-you?

pass "What's up, Doc?/Not much; you?/Not much; you?:I-coded-userdata-today"

- Enter the account information that contains blanks by enclosing the entire account information parameter in quotation marks. You can use single or double quotation marks. If the account information itself contains a quotation mark, use the other style of quotation marks to enclose the password phrase.

Example: Enter the account information *What's up, Doc?* as *"What's up, Doc?"*, but not as *'What's up, Doc?'*.

- Do not use quotation marks to enclose a *password/newpass/newpass:userdata* parameter sequence that is comprised only of any of the following characters:
 - Uppercase or lowercase letters
 - Numerals from 0 to 9
 - The following special characters:
 - @
 - #
 - \$
 - -
 - {
 - .
 - (

-)
- *
- %
- +

This rule applies also to *account_information*.

Example: Enter the password phrase *JoeIBMer@ibm.com* as *JoeIBMer@ibm.com*, but not as *'JoeIBMer@ibm.com'*, nor as *"JoeIBMer@ibm.com"*.

Restrictions:

- A password, password phrase, or the account information that you enter at the z/OS FTP client must not contain both single quotation mark and double quotation mark characters. You can use either style of quotation marks in the password, password phrase, or account information, but not both.

Example: The password phrase *What's up, Doc?* is valid because it contains only single quotation marks. You enter it at the z/OS FTP client as *"What's up, Doc?"*. The password phrase *"What's up, Doc?"* with the double quotation marks as part of the password phrase cannot be entered at the z/OS FTP client because it contains both styles of quotation marks.

- When entering this subcommand in a USS environment, you can enter only up to 510 characters including the subcommand name. When entering the arguments *password/newpass/newpass:userdata account_information*, such that *password* and *newpass* are password phrases, you must take this into account.

Related Topic: See [“User subcommand—Identify yourself to a host or change your TSO user ID password”](#) on page 293 for more information.

PRivate subcommand—Set the protection level for data transfers to PRIVATE

Purpose

Use the PRivate subcommand to set the protection level for data transfers on the data connections to private. This subcommand is equivalent to specifying the PROtect PRivate subcommand.

Format

➡ PRivate ➡

Parameters

There are no parameters for this subcommand.

Examples

To set the protection level to private, enter:

```
private
```

Usage

- This subcommand is not valid when there is no active security mechanism.
- Data transmissions are confidentiality and integrity protected by encryption.

PROMpt subcommand—Toggle interactive prompting for M* commands

Purpose

Use the PROMpt subcommand to toggle interactive prompting for MDelete, MGet, and MPut commands. Prompting is the default action unless the FTP session was started with the **-i** option, which turns off interactive prompting.

Format

➡ PROMpt ➡

Parameters

There are no parameters for this subcommand.

Example

The following example shows using the MPut command with interactive prompting on.

```
Command:
mput file*
Mput FILE1 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,128
200 Port request OK.
>>>STOR FILE1
125 Storing data set /u/user31/temp/FILE1
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds. Transfer rate 16.40 Kbytes/sec.
Mput FILE2 (Yes|No|Quit|Stop prompting)? no
Mput FILE3 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,129
200 Port request OK.
>>>STOR FILE3
125 Storing data set /u/user31/temp/FILE3
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds. Transfer rate 16.40 Kbytes/sec.
Command:
```

Context


See [“Using FTP” on page 19](#) for more information about the **-i** option.

PROtect subcommand—Set the protection level for data transfers

Purpose

Use the PROtect subcommand to set the protection level for data transfers on the data connections.

Format

➡ PROtect 

Parameters

CLEAR

Data transmissions are not protected. Specifying PROtect CLEAR is equivalent to specifying the CLEar subcommand.

PRIVATE

Data transmissions are confidentially and integrity protected. Specifying PROtect PRIVATE is equivalent to specifying the PRIvate subcommand.

SAFE

Data transmissions integrity are protected by cryptographic checksum.

Examples

To set the protection level to private, enter:

```
prot private
```

Usage

This subcommand is not valid when there is no active security mechanism.

PROXy subcommand—Execute FTP subcommand on secondary control connections

Purpose

Use the PROXy subcommand to execute an FTP subcommand on secondary control connections. PROXy enables the FTP client to connect simultaneously to two remote FTP servers and then to establish a data connection between the two servers for the purpose of transferring files between those servers.

Format

►► PROXy — *subcommand* ◄◄

Parameters

subcommand

The name of any FTP subcommands except those listed in the first note in [“Usage” on page 237](#). The first PROXy *subcommand* should be Open, which establishes the secondary server connection.

The following subcommands behave differently when prefaced by the PROXy subcommand:

- Open establishes the secondary server connection.
- CClose closes the secondary server connection.
- Get and MGet transfer files from the host on the primary connection to the host on the secondary connection.
- PUt, MPut, and AAppend transfer files from the host on the secondary connection to the host on the primary connection.

Examples

The following example shows a proxy open to establish connection to a secondary server.

```

Command:
proxy open 9.67.113.57 6321
Connecting to: 9.67.113.57 port: 6321.
220-FTPDJG1 IBM FTP CS V1R4 at MVS164, 13:06:23 on 2003-01-14.
220 Connection will not timeout.
NAME (9.67.113.57:USER33): user34
>>>USER user34
331 Send password please.
PASSWORD:
>>>PASS
230 USER34 is logged on. Working directory is "USER34.".

```

The following example shows the commands for a proxy between IPv4 nodes:

- PASV to the secondary server
- PORT to the primary server
- RETR to the primary server
- STOR to the secondary server

Two 250 replies are received by the client, one from each server.

```

Command:
proxy m1 mx
>>>PASV
227 Entering Passive Mode (9,67,113,57,5,121)
>>>PORT 9,67,113,57,5,121
200 Port request OK.
>>>RETR m1
125 Sending data set /u/user33/mpp1/m1
>>>STOR mx
125 Storing data set USER34.MX
250 Transfer completed successfully.
250 Transfer completed successfully.
Command:

```

The following shows a proxy open to establish connection to a secondary server with an IPv6 address:

```

Command:
proxy open local167v6
Connecting to: Local167v6 2001:0DB8:c2d4::9:67:115:12 port: 21.
220-Welcome to my test system.
220-You are logged on from 2001:0DB8:c2d4::9:67:115:13
220 Connection will not timeout.
Command:
user user2
>>> USER user2
331 Send password please.
PASSWORD:

>>> PASS
230 USER2 is logged on. Working directory is "/".

```

This example shows the commands for a proxy PUt between IPv6 nodes:

- EPSV to the secondary server
- EPRT to the primary server
- STOR to the primary server
- RETR to the secondary server

As in the first example, two 250 replies are received by the client.

```

Command:
proxy put bob testfile
Load module transfer does not support load module rename
>>> EPSV
229 Entering Extended Passive Mode (|||1027|)
>>> EPRT |2|2001:0DB8:c2d4::9:67:115:12|1027|
200 EPRT request OK
>>> STOR testfile

```

```
125 Storing data set /tmp/myTest/testfile
>>> RETR bob
125 Sending data set /tmp/myTest/bob
250 Transfer completed successfully.
250 Transfer completed successfully.
Command:
```

Usage

- The following subcommands are not valid proxy subcommands:
 - DEBug
 - DUMP
 - DELimit
 - GLob
 - LANGuage
 - LCd
 - LMkdir
 - LOCStite
 - LOCStat
 - LPwd
 - PROMpt
 - QUIt
 - REStart
 - SENDPort
 - SENDSite
 - SRestart
 - SUnique
 - TSO
 - Verbose
- To receive help from a server on a secondary control connection, enter PROXy HElp SERVER.
- Data transfer in PROXy mode can be restricted if the server is set up to reject PORT and EPRT commands with certain parameters. See the [z/OS Communications Server: IP Configuration Guide](#) for more details.
- If an open subcommand is entered as a proxy subcommand by the client and the session is currently protected by a security mechanism (for example, TLS), then the subcommand is rejected with the following message:

```
Proxy open is not supported with security mechanisms
```

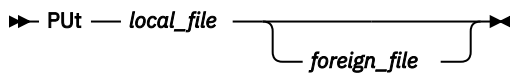
- If the connection to one server is IPv4 and the connection to the other is IPv6, proxy transfers might not be possible. The two servers must have the ability to connect to each other as well as to the client. A z/OS FTP server cannot be the primary server in a proxy transfer unless the connection from the client to each of the servers is of the same protocol.
- Data transfer in PROXy mode can be restricted if the server is set up to reject the redirection of the passive (PASV) data connection using PASSIVEDATACONN NOREDIRECT. See the [z/OS Communications Server: IP Configuration Reference](#) for more details.

PUt subcommand—Copy data sets to the remote host

Purpose

Use the PUt subcommand to copy data sets from your local host to the remote host.

Format



Parameters

local_file

Specifies the name of the file on your local host being sent to the remote host.

foreign_file

Specifies the name that the delivered data set is given on the remote host. If the *foreign_file* name is not specified, the *foreign_file* name is the same as the *local_file* name.

Restriction: If FTP does not support directory content transfers in partitioned data sets, it is not possible to FTP load modules.

Results:

- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the PUt subcommand, FTP might truncate data records and you might lose data, if one of the following occurs:
 - If you are creating a new file at the server and the value of LRecl, as shown by the STatus subcommand, is a value less than the LRecl of the transmitted data set and SENDSite subcommand has been set to OFF, then FTP truncates the transmitted data set.
 - If the data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, then FTP truncates the transmitted data set.
- When a PUt is issued, FTP automatically sends a SItE subcommand containing record format information for the file or data set. To toggle this off, you must first issue a SENDSite subcommand. See [“SENDSite subcommand—Toggle the sending of site information”](#) on page 246 for more detailed information.
- If the remote host already has a file with the name specified by *foreign_file*, the remote host overwrites the existing file. If the remote host does not have a file with the same name specified by *foreign_file*, the remote host creates a new file.
- When a PDS or PDSE member is transmitted, the user data associated with the PDS member is also transferred to the directory on the target host if the following are true:
 - Data is in block or compressed data transfer mode
 - Data has a representation type of EBCDIC
 - Transfer is from one MVS directory to anotherNo PDS directory information is transferred if the member is null (empty).
- If the remote FTP server is a z/OS FTP server that is V1R8 or later, UNIXFILETYPE=FIFO is configured at the remote host, and the foreign file directory is in the z/OS UNIX file system, the following apply:
 - The remote host creates a new named pipe if a file with that name does not already exist.
 - The remote host appends the local file to the foreign file if the foreign file exists as a named pipe.
 - The remote host rejects the transfer if the foreign file exists as a regular z/OS UNIX file.

- The remote host rejects the transfer if the storage method is store-unique. Use the SUnique subcommand to change the storage method.

For more information about using z/OS UNIX named pipes, see [“Using z/OS UNIX System Services named pipes”](#) on page 104.

- If the local file is a z/OS UNIX named pipe, the following apply:
 - You must configure UNIXFILETYPE FIFO to send data from the named pipe.
 - Sending a named pipe permanently removes the data from the named pipe.
 - FTP cannot read from the named pipe until another process on the client host opens the named pipe for writing. The z/OS FTP client waits up to the number of seconds specified by the FIFOOPEN TIME value for another process to open the named pipe. If a process does not open the named pipe, the client fails the file transfer.
 - FTP waits up to the number of seconds specified by the FIFOIO TIME value for each read from the named pipe to complete. If the client cannot read any data from the named pipe for the number of seconds specified by the FIFOIO TIME value, it fails the file transfer.

Requirements:

- To put files on the remote host with unique file names, you must have set unique storage on before issuing the P Ut command. Use the SUnique subcommand to change the storage method.
- To send a data set to the remote host, you must have a defined working directory on the remote host and write privileges to the files in this working directory.

UPDATE authority is the minimum required for write access. To ensure that the newly created data set can be written to before the data set has been allocated and opened, FTP validates that the user ID has at least UPDATE authority. If this fails, then FTP will be able to issue a reply to the client which is indicative of the failure. If open is allowed to continue and it fails due to lack of authority, then the reply will not be as definitive.

Related topics:

- See [“SUnique subcommand—Changes the storage method ”](#) on page 286 for information about changing the storage method on the remote host.
- See Appendix A, [“Specifying data sets and files,”](#) on page 359 for more information about naming conventions.
- P Ut can be used with the PROXy subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See [“PROXy subcommand—Execute FTP subcommand on secondary control connections”](#) on page 235 for more information.
- See UNIXFILETYPE, FIFOIO TIME, and FIFOOPEN TIME statements in the using z/OS UNIX named pipes details in [z/OS Communications Server: IP Configuration Reference](#) for more information.

PWd subcommand—Display the current working directory

Purpose

Use the PWd subcommand to display the name of the current working directory on the remote host.

Format

➤ PWd ➤

Parameters

There are no parameters for this subcommand.

Examples

Display the name of the current working directory:

```
pwd
>>>PWD
257 "'USER17.HSMTEST.'" is working directory
Command:
```

Display the name of the current z/OS UNIX file system working directory:

```
pwd
>>>PWD
257 "/u/user121/example" is the HFS working directory.
Command:
```

QUIt subcommand—Leave the FTP environment

Purpose

Use the QUIt subcommand to disconnect from the foreign host and end the FTP session.

Format

➤ QUIt ➤

Parameters

There are no parameters for this subcommand.

Usage

- The QUIt subcommand ends the FTP session with the remote host and exits FTP on the local host. To establish a new session, use the FTP command.
- In a z/OS UNIX environment, you can also press Ctrl-C to end an FTP session.
- When running with both a primary and a secondary server (by using the PROXy subcommand), the QUIt subcommand disconnects both sessions.

Context

See [“Using FTP” on page 19](#) for information about the FTP command.

QUOte subcommand—Send an uninterpreted string of data

Purpose

Use the QUOte subcommand to send an uninterpreted string of data to the server port on the foreign host.

The QUOte subcommand bypasses the FTP interface of your local host. You can use the QUOte subcommand to send commands that the remote server understands, but that the local host does not understand.

Format

➤ QUOte — *string* ➤

Parameters

string

Specifies the data to be sent verbatim to the remote host FTP server.

Examples

- For example, QUOTE TYPE B 1 causes the FTP server to change its transfer type to Shift JIS kanji, without changing the transfer type in the FTP client. The client in this example should be set to the ASCII transfer type before the QUOTE subcommand is issued.
- The following example shows the screen display when setting the DBCS transfer type to JIS78KJ, shift-in JISROMAN, and then setting it to HANGEUL using EBCDIC SO/SI characters. The example shows an MVS TCP/IP FTP client connected to an MVS TCP/IP FTP server. All three methods of setting the DBCS transfer type are shown.

```
User: jis78kj (jisroman
System: >>>TYPE b 4 r
        200-Representation type is kanji JIS 1978 shift-in JISROMAN
        200 Standard DBCS control used
        Command:
User: type b 4 r
System: >>>TYPE b 4 r
        200-Representation type is kanji JIS 1978 shift-in JISROMAN
        200 Standard DBCS control used
        Command:
User: jis78kj (jisroman notype
System: Command:
User: quote type b 4 r
System: >>>type b 4 r
        200-Representation type is kanji JIS 1978 shift-in JISROMAN
        200 Standard DBCS control used
        Command:
User: hangeul (sosi ebcdic
System: >>>TYPE b 5 s e
        200-Representation type is Hangeul
        200-SO/SI characters X'0E'/X'0F' used
        200 Data transfer is mixed SBCS/DBCS
        Command:
User: type b 5 s e
System: >>>TYPE b 5 s e
        200-Representation type is Hangeul
        200-SO/SI characters X'0E'/X'0F' used
        200 Data transfer is mixed SBCS/DBCS
        Command:
User: hangeul (sosi ebcdic notype
System: Command:
User: quote type b 5 s e
System: >>>type b 5 s e
        200-Representation type is Hangeul
        200-SO/SI characters X'0E'/X'0F' used
        200 Data transfer is mixed SBCS/DBCS
        Command:
```

Usage

- No parsing or validity checking is performed on the character string you enter by FTP on your local host. If the character string you send to the FTP server is part of a required sequence of commands, you are required to provide this sequence correctly, or the results are unpredictable.
- The QUOTE subcommand can be used to generate any of the DBCS TYPE commands supported by the server. This subcommand is used when the FTP server supports the DBCS TYPE command, but the FTP client does not.

RECORD subcommand—Set the file structure to record

Purpose

Use the REcord subcommand to set the file structure to record. This is equivalent to specifying the STRucture R subcommand. See [“STRucture subcommand—Set the file structure”](#) on page 286 for more information.

Format

➤ REcord ➤

Parameters

There are no parameters for this subcommand.

REName subcommand—Rename files

Purpose

Use the REName subcommand to rename a file, data set, or z/OS UNIX named pipe on the remote host.

Format

➤ REName — *original_name* — *new_name* ➤

Parameters

original_name

Specifies the current name of the file.

new_name

Specifies the new name of the file.

Results:

- For MVS data sets, if the data set that is specified by the *new_name* value already exists, the server rejects the rename request.
- For z/OS UNIX files and named pipes, if the file that is specified by the *new_name* value already exists, the existing file is replaced.

Restriction: When you use the FTP RENAME subcommand with a generation data group (GDG), for example, `RENAME SOURCE.FILE MY.GDG(+1)`, serialization of the GDG data set is not assured. To avoid this, instead use the FTP PUT subcommand. For more information about GDG processing, see Information APAR II08285.

REStart subcommand - Restart a checkpointed data transfer

Purpose

Use the REStart subcommand to restart a check pointed file or data set transfer that has been interrupted.

Format

➤ REStart ➤

Parameters

There are no parameters for this subcommand.

Requirements:

- To restart a file transfer with the REStart subcommand, you must have had check pointing enabled during the file transfer you want to restart.
- Before you issue the REStart subcommand, set up the same file transfer environment (such as file transfer mode, type, and CHPTPREFIX file or data set) that you had configured during the file transfer that you want to restart.

Guidelines:

- Use the REStart subcommand to resume file transfer when a check pointed file transfer request fails because of a temporary condition such as the loss of the connection between the client and the server.
- Enable check pointing by the following steps:
 - Configure a check point data set or file, and a check point interval greater than zero. FTP uses the check point data set or file to store the information that it needs to resume the data transfer. The check point interval determines how often the client and server exchange information needed to restart the file transfer.
 - Transfer files and data sets with type EBCDIC and mode block or compressed. Check pointing is available only for type EBCDIC file transfers in block or compressed mode.

Rule: Do not enable check pointing if the server you are logged into does not support the REST command.

By default, check pointing is enabled in both directions of file transfer when you enable file transfer. You can control whether check pointing is enabled for Get subcommand processing by configuring RESTGET at the FTP client. You can use the LOCStite subcommand, or code the RESTGET statement in FTP.DATA, to configure RESTGET.

If you are logged into the z/OS FTP server, you can control check pointing at the server with the server CHKPTINT and RESTPUT configuration options.

- Every time when you start a new file transfer while check pointing is enabled, FTP reuses the check point file or data set. To prevent losing restart information after a failed or interrupted file transfer, do one of the following steps before transferring another file or data set:
 - Issue the REStart subcommand.
 - Save the check point file or data set. You will have to restore the check point file or data set before issuing the REStart subcommand.

Restriction: Do not edit the check point file or data set.

- If you transfer two or more data sets simultaneously with check pointing enabled, assign each session a different check point data set to prevent two users from contending for the same check point file or data set.

Results

- The REStart subcommand restarts the last checkpoint file transfer request at the point of the last valid checkpoint stored in the checkpoint data set.
- After a successful file transfer with check pointing enabled, FTP deletes the check point file or data set.
- The LOCStite NORESTGet subcommand prevents opening the checkpoint data set for a Get request.
- The MVSGet or MVSPut subcommand supports checkpointing for block mode restart of an interrupted file transfer only for physical sequential data sets. The MVSGet and MVSPut subcommands do not support checkpointing for block mode restart of PDS or library data sets.

Related topics:

For information about configuring the check point file or data set, see CHKPTPrefix in [“LOCStite subcommand—Specify site information to the local host” on page 179](#) or the [CHKPTPREFIX \(FTP client\) statement information in z/OS Communications Server: IP Configuration Reference](#).

For more information about configuring the check point interval, see [CHKptint](#) (SItE subcommand), [CHKptint](#) (LOCStE subcommand), or [CHKPTINT](#) (FTP client and server) statement in [z/OS Communications Server: IP Configuration Reference](#).

For more information about the RESTPUT configuration option, see “[SItE subcommand—Send site-specific information to a host](#)” on page 247 or the [RESTPUT](#) (FTP server) information in [z/OS Communications Server: IP Configuration Reference](#).

RMdir subcommand—Remove a directory on the remote host

Purpose

Use the RMdir subcommand to remove a directory on the remote host.

Format

➤ RMdir — *directory* ➤

Parameters

directory

Specifies the name of the directory to be removed.

Usage

- The RMdir subcommand sends a request to the remote host FTP server to remove a directory with name *directory* from the current remote directory.
- The RMdir subcommand can be used to delete a PDS.

SAfe subcommand—Set the protection level to safe

Purpose

Set the protection level on data transfers to "safe". Data transmissions are integrity-protected by cryptographic checksum.

Format

➤ SAfe ➤

Parameters

There are no parameters for this subcommand.

SChinese subcommand—Change the data transfer type to SCHINESE

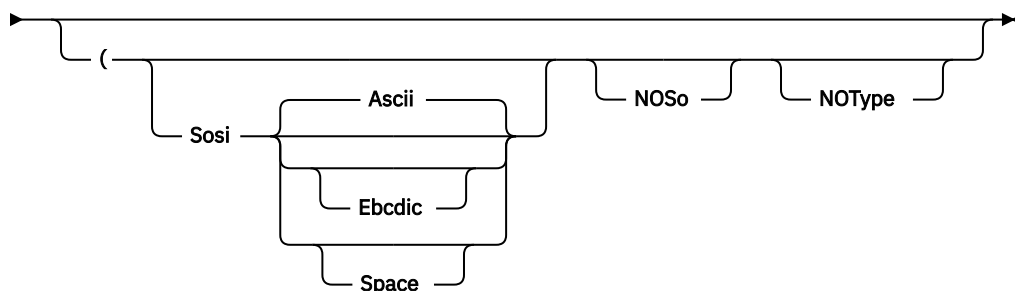
Purpose

Use the SChinese subcommand to change the data transfer type to Simplified Chinese.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the [SItE/LOCStE SBDataconn](#) or [SItE XLate](#) subcommand to have the server (or client) change the SBCS translation for the data connection.

Format

➤ SChinese ➡



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by the one of the following parameters — Ascii, Ebcdic, or Space. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The SChinese client subcommand is equivalent to the TYPE B 9 server command.

Context

See [“FTP with traditional DBCS support”](#) on page 78 and [“Support for MBCS languages”](#) on page 81 for more information.

SENDPort subcommand—Toggle the sending of port information

Purpose

Use the SENDPort subcommand to toggle the automatic sending of the PORT command.

Format

➤ SENDPort ➡

Parameters

There are no parameters for this subcommand.

Usage

- By default, the SENDPort subcommand is turned on when you start an FTP session. Each time you use the SENDPort subcommand, it is turned alternately on and off.
- FTP does not send PORT commands for data transfer when you disable PORT commands by toggling the function off.
- SENDPort is useful for communication with those FTP implementations that ignore PORT commands, but show (incorrectly) that the PORT command has been accepted.
- To determine if the sending of port information is enabled or disabled on your local host, use the LOCStat subcommand.
- The sendport setting is ignored during proxy transfer.

Context

See [“LOCStat subcommand—Display local status information ” on page 201](#) for more information about LOCStat subcommand.

SENDSite subcommand—Toggle the sending of site information

Purpose

Use the SENDSite subcommand to toggle the automatic sending of the SITE commands when sending a data set to a foreign host.

Format

➡ SENDSite →

Parameters

There are no parameters for this subcommand.

Usage

- By default, the SENDSite subcommand is turned on when you start an FTP session. Each time you use the SENDSite subcommand, it is turned alternately on and off.

When turned on, FTP sends a SITE command that contains record format information for the file or data set when you issue the PUt or MPut subcommand.
- SENDSite is useful when you want to PUt a file to the remote host and have the file created with the same characteristics as defined at the local host.
- If you are using either an SMS data class or a model DCB at your MVS server to provide the logical record length or record format, you must toggle the SENDSite setting off at the client. Otherwise, the Site information that is sent automatically by the client overrides the values provided by the SMS dataclass or model DCB.
- To determine if the sending of site information is enabled or disabled on your local host, use the LOCStat subcommand.
- The sendsite setting is always ignored if you issue a mkdir subcommand with the (like parameter. The client must send a SITE command to the server to set the server's site variables before allocating the directory.

Context

- See “[LOCStat subcommand—Display local status information](#)” on page 201 for information about the LOCStat subcommand.
- See “[MKDir subcommand—Create a directory on the remote host](#)” on page 216 for information about the MKDir subcommand.

SItE subcommand—Send site-specific information to a host

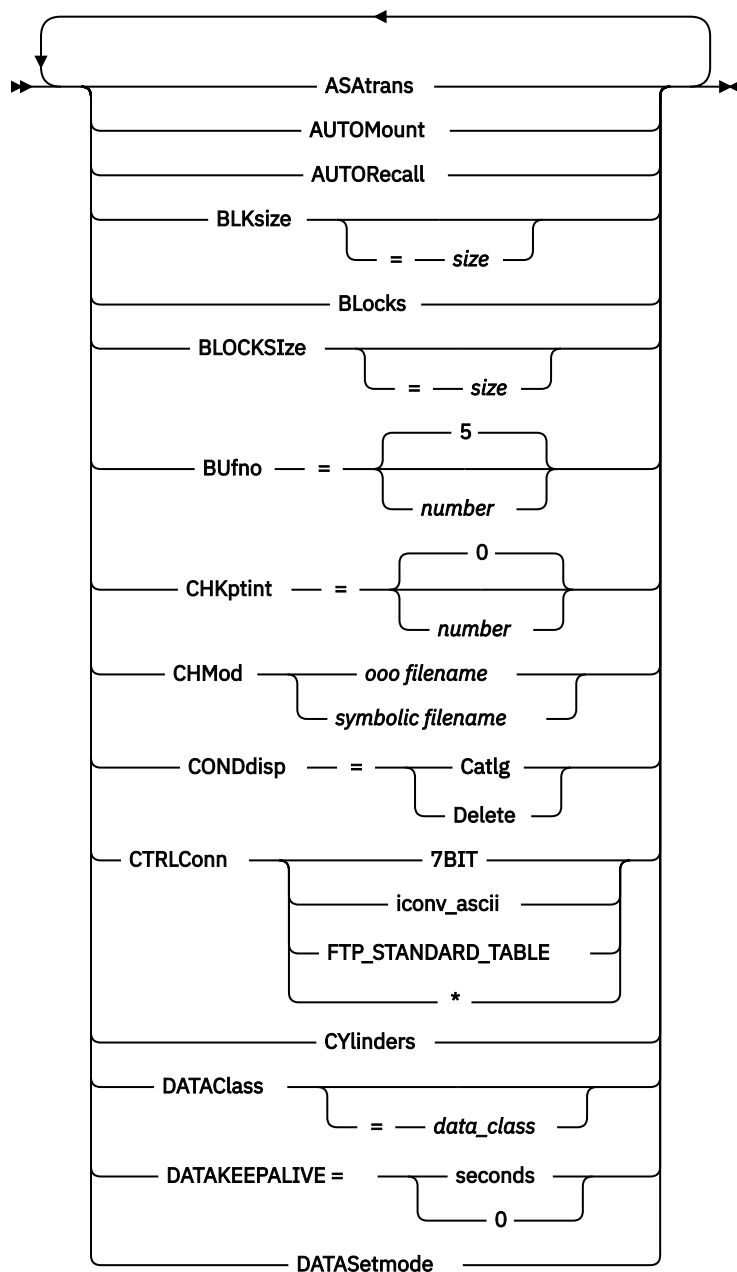
Purpose

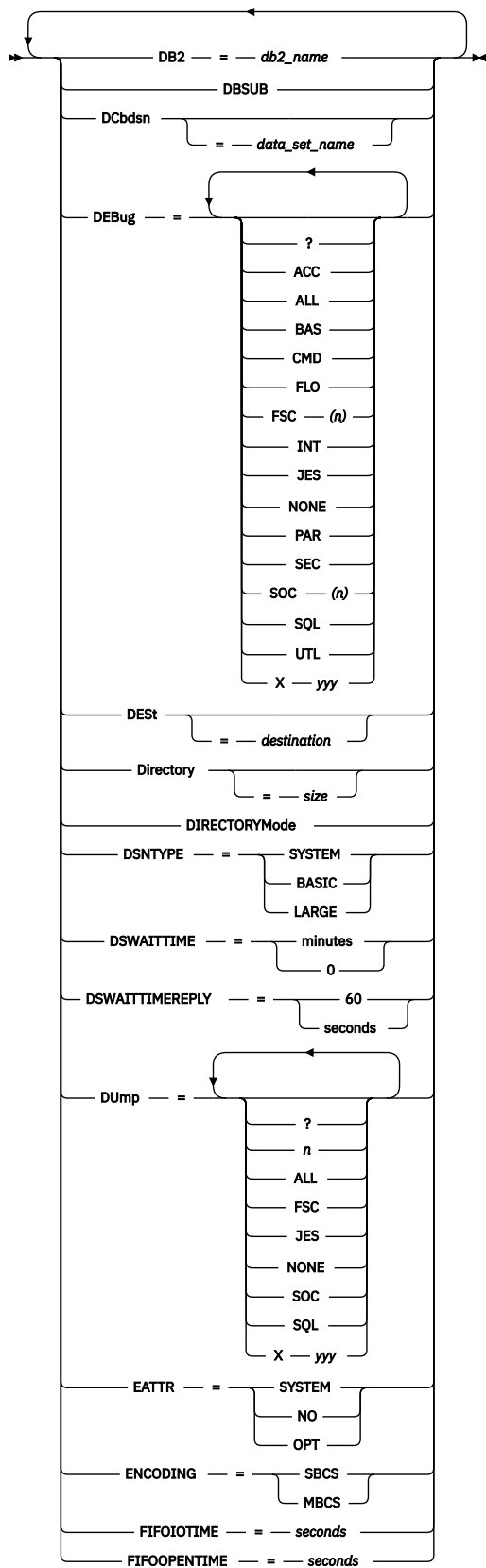
Use the SItE subcommand to send information that is used by the remote host to provide services specific to that host system.

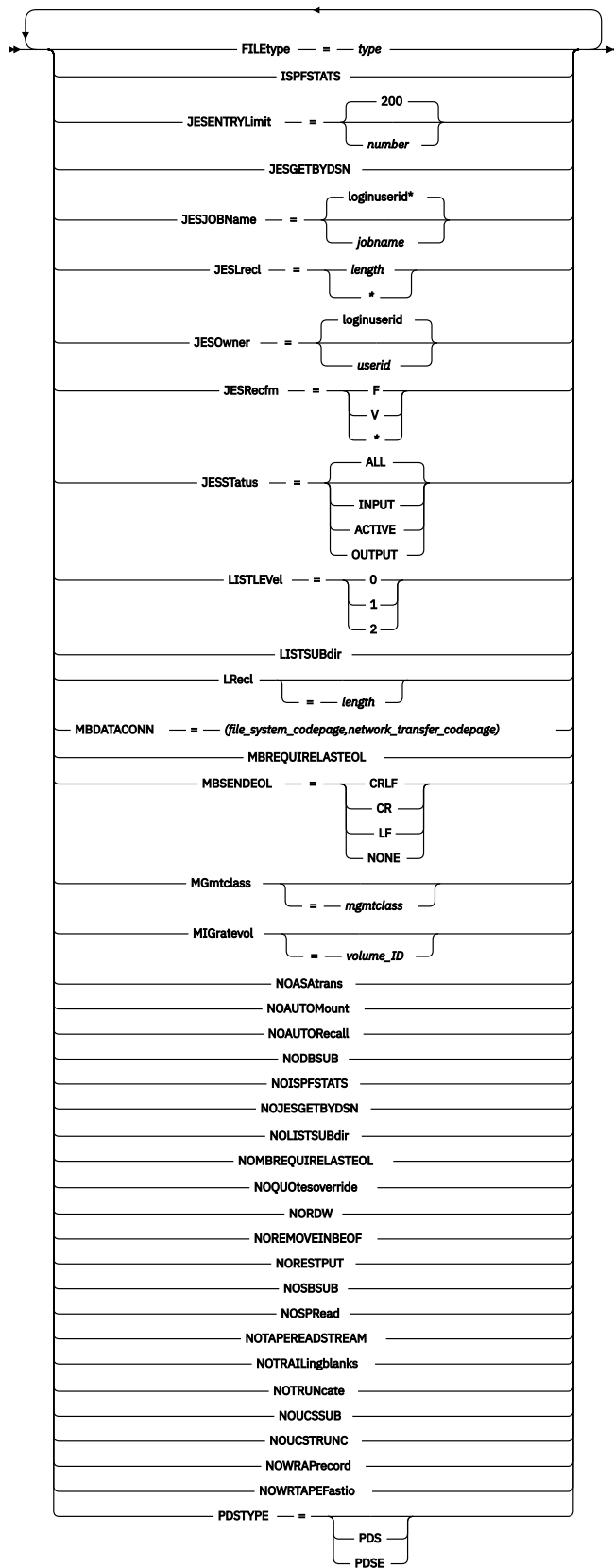
Format

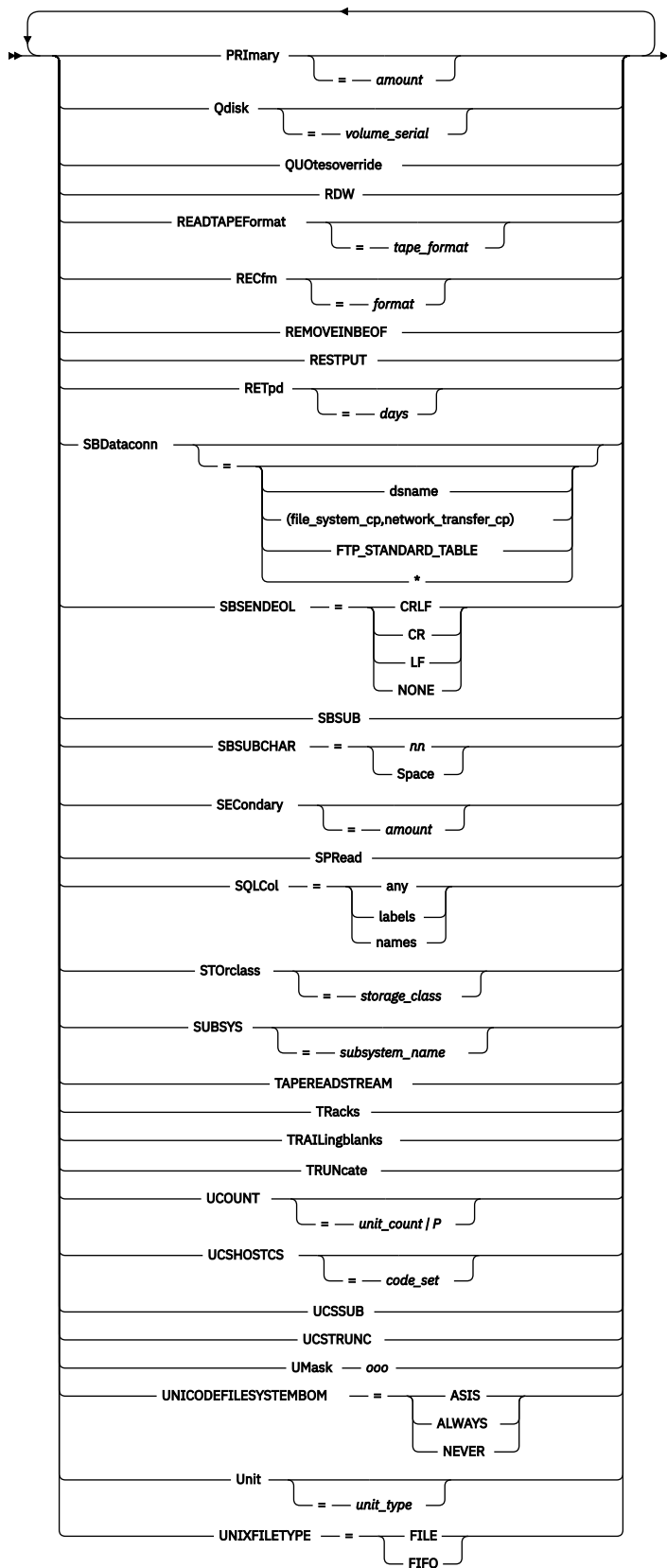
►► SItE — options ◄◄

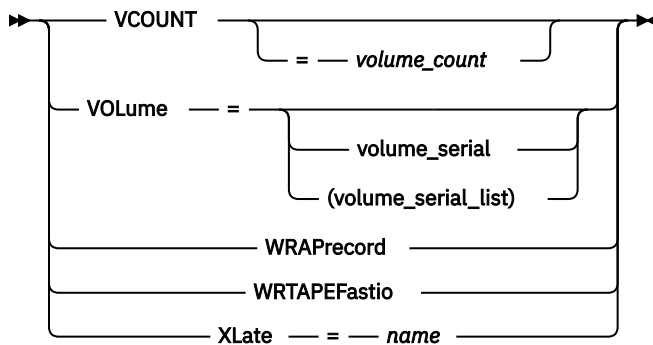
options











Parameters

ASAtans

Permits the FTP server to interpret the characters in the first column of ASA files being transferred as print control characters.

AUTOMount

Permits automatic mounting of volumes for data sets on volumes that are not mounted. If AUTOMount is specified and an unmounted volume is needed, a message is automatically issued to the MVS operator console requesting that the volume be mounted. The MVS operator must then mount the volume and reply to the message before FTP can proceed.

AUTOREcall

Permits automatic recall of migrated data sets.

BLKsize

Specifies the block size of a newly allocated data set. BLKsize is functionally equivalent to BLOCKSize. The BLOCKSize parameter is obsolete, but it is accepted to provide compatibility with previous releases of z/OS TCP/IP.

When specified without a size, no block size is used when allocating the new data set. When specified without a size, the equal sign (=) is optional.

Specify BLKsize with no value if you are also specifying DATAclass=*data_class* and you want the SMS data class to provide the BLKsize value, or if you are specifying DCbdsn=*data_set_name* and you want to use the block size from the DCBDSN data set. If BLKsize=*size* is specified with either the DATAclass or DCbdsn parameters, the value specified by the SITE BLKsize parameter overrides the DATAclass or DCbdsn block size.

Notes:

1. If you specify BLKsize without a size or with a size of 0, FTP does not specify the block size when allocating new data sets.
2. Be especially careful specifying both BLKsize= and BLOCKs. While there are conditions where this is tolerated, if a valid BLKsize cannot be determined, the data set will not be created when the allocation is attempted.

size

Specifies the block size of a newly allocated data set. The valid range is 0–32 760.

BLKsize=0 is a special case. When BLKsize=0 is specified, the operating system attempts to determine a block size for the new data set. FTP does not create the new data set unless the system is able to establish a nonzero block size.

BLOCKs

Specifies that primary and secondary space allocations are in blocks.

If both PRImary and SECondary are unspecified, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the BLOCKs parameter is ignored.

BLOCKSize

Specifies the block size of a newly allocated data set. BLOCKSize is functionally equivalent to BLKsize. BLOCKSize is obsolete but it is accepted to provide compatibility with previous releases of z/OS TCP/IP. See the BLKsize parameter for more information.

BUfno

Specifies the number of access method buffers that is used when data is read from or written to a data set. The default value is 5.

number

The number of buffers used. The valid range is 1–35.

CHKptint

Specifies the checkpoint interval for the FTP server. The checkpoint interval applies when the server processes the RETR command.

The checkpoint interval is the number of records that are sent between restart markers when you transfer files in EBCDIC block mode or EBCDIC compress mode. If the checkpoint interval is 0, no checkpointing occurs and no restart markers are transmitted. The default value is 0.

Results: A CHKptint value that is greater than 0 enables checkpointing for outbound file transfers from the server that meet the following conditions:

- Type must be EBCDIC
- Mode must be block or compressed
- Filetype must be SEQ

Checkpointing never occurs when the server file is a z/OS UNIX named pipe.

number

The checkpoint interval for the sending site in a file transfer request. This value is used to determine when checkpoint marker blocks are to be transmitted so that transmission can be restarted based on the information in the last marker.

A large checkpoint interval means that a large amount of data is sent in between markers and few markers are sent. A smaller checkpoint interval means that less data is sent between markers and more markers are sent.

The cost involved with using a nonzero checkpoint interval is the markers themselves are transmitted, which means that more bytes are being sent across the network (approximately 44 bytes per marker).

To estimate the appropriate checkpoint interval, use the following formula. You need to know the record length of the file you are transferring and how much data you think can be transmitted reliably.

$$\text{CHKPTINT} = \frac{\text{amount of data in interval}}{\text{record length of the file}}$$

Do not use a CHKptint more often than once every 200 KB of data transferred. The following is an example of a file that you are transferring with 80-byte records in which the checkpoint interval is 2560:

$$\begin{aligned}\text{CHKPTINT} &= 200\text{KB} / 80 \text{ bytes} \\ &= 200 * 1024 \text{ bytes} / 80 \text{ bytes} \\ &= 2560\end{aligned}$$

CHMod

Changes the permission bits for a file.

ooo filename

ooo is an octal mask representing the permissions you want to assign to filename. Form the octal mask by OR'ing the constants corresponding to the permission bits you want set:

| | |
|------------|----------------------------------|
| 400 | User read |
| 200 | User write |
| 100 | User execute (or list directory) |
| 040 | Group read |
| 020 | Group write |
| 010 | Group execute |
| 004 | Other read |
| 002 | Other write |
| 001 | Other execute |

You cannot use the SITE subcommand CHMod parameter to set the following permission bits:

- Set-user-ID bit
- Set-group-ID bit
- Sticky bit

See the [z/OS UNIX System Services User's Guide](#) and the [z/OS UNIX System Services Command Reference](#) for more information about file permissions.

symbolic filename

symbolic represents the permissions you want to apply to *filename*.

Note: *symbolic* is specified as follows:

```
{u|g|o|a}{=|+|-}{r|w|x|rwx|wrx|wx}
```

where u, g, o, a, =, +, -, r, w, and x are as defined for the z/OS UNIX chmod command.

If *filename* does not begin with a slash character (/), it is appended to the current working directory. If *filename* does begin with a slash character (/), it is interpreted as a complete directory name.

The file name specified must be a z/OS UNIX file name for a single file and cannot contain a wildcard (*) for multiple files. The setting of QUOTESoverride is ignored and all quotation marks are treated as part of the file name.

The CHMOD keyword must be the only keyword or last keyword on a SITE subcommand.

CONDdisp

Specifies the disposition of the data set if a store operation for a new data set ends before all of the data is written.

Catlg

Specifies that a data set is kept and cataloged when an FTP file transfer ends prematurely.

Delete

Specifies that a data set is deleted when an FTP file transfer ends prematurely.

Delete is ignored if the file transfer failed as a result of the FTP server being terminated or if the server has received checkpoint information during data transfer.

CTRLConn

Specifies the ASCII code page to be used for control connections. The valid subcommands are:


```
SITE CTRLConn=7BIT
SITE CTRLConn=iconv_ascii
SITE CTRLConn=FTP_STANDARD_TABLE
SITE CTRLConn=*
```

See [“Support for SBCS languages” on page 77](#) for more information.

7BIT

Indicates 7-bit ASCII is to be used.

iconv_ascii

A name recognized by iconv to indicate an ASCII code page. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

FTP_STANDARD_TABLE

Specifies that the FTP internal tables, which are the same as the tables that are shipped in TCPXLBIN(STANDARD), are to be used on the control connection.

Specifies that the ASCII used at initialization is to be used.

Note: Setting the control connection code page with SITE CTRLCONN disables UTF-8 encoding. You must start a new session to restore UTF-8 encoding.

CYLinders

Specifies that primary and secondary space allocations are in cylinders.

If both PRIMary and SECondary are unspecified, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the CYLinders parameter is ignored.

DATAclass

Used to specify the SMS data class, as defined by your organization, for the target host. Specifying DATAclass with no parameter value cancels the data class specification. The equal sign (=) is optional in this case.

See [“Specifying values for new data sets” on page 82](#) for more information about specifying attributes when allocating new data sets.

data_class

Specifies the SMS data class, as defined by your organization, for the target host. If values are specified for any of the following SITE parameters, the values specified by the SITE parameter override the value specified in the SMS data class:

- BLKsize
- Directory
- LRecl
- PRIMary
- RECfm
- RETpd
- SECondary

If the DCbdsn SITE parameter is specified, the LRecl, RECfm, BLOCKSsize, and RETpd (if specified) of the DCBDSN data set overrides the values specified in the data class.

If the MGmtclass site parameter is specified, and the requested management class specifies a retention period, the retention period value of the management class can override the retention period value of the data class.

A SITE DATAclass command pointing to an SMS data class must be used to create a PDSE. A load module loading from a temporary data set will always be a REPLACE operation, in that existing members will be overwritten in case of name conflicts. LMTR will not be performed in STOU mode (when SUnique is turned on).

DATAKEEPAIVE

Specifies the data connection keepalive timer value for the FTP server.

seconds

The number of seconds that elapse before a keepalive packet is sent on the FTP data connection. Valid values are in the range 60-86400 or 0. If you specify the value 0, the DATAKEEPAIVE timer is disabled. For active mode data connections, the keepalive timer that is configured in the PROFILE.TCPIP controls how often keepalive packets flow on the data connection. For passive mode data connections, FTP suppresses the PROFILE.TCPIP keepalive timer.

Result: Specifying a DATAKEEPAIVE value prevents a network device from closing the data connection during periods of inactivity on the data connection.

DATASetmode

Specifies that all the data set qualifiers below the current directory are treated as entries in the directory (disables DIRECTORYMode).

DB2

Specifies the name of the Db2 subsystem.

db2_name

The name of the Db2 subsystem.

DBSUB

Specifies that substitution is allowed for data bytes that cannot be translated in a double-byte character translation. The substitution character is selected by the C/C++ iconv() function; see the information about Locales and Character Sets in [z/OS XL C/C++ Programming Guide](#) for more details.

DCbdsn

Specifies the name of the MVS data set to be used as a model for allocation of new data sets. Specifying DCbdsn with no parameter value cancels the DCbdsn specification.

data_set_name

Specifies the name of the data set. The file name must be an MVS data set name. z/OS UNIX file names are not allowed on the DCbdsn parameter. The setting of QUOTESoverride is ignored. If the file name is enclosed in single quotation marks, it overrides the current working directory; otherwise it is appended to the current working directory.

Notes:

1. Specify SItE RECFm, LREcl, and BLKsize parameters with no values to allow characteristics from the model DCB to be used.
2. To override the model characteristics of RECFm, LREcl, BLKsize, or RETpd, specify a value on the SItE command.
3. Ensure that SENDSite subcommand is toggled off. Otherwise, the SItE information that is sent automatically by the client overrides the values provided by the model DCB.
4. If MGmtclass is specified, the RETpd value of the MGmtclass can override the RETpd value.

Specifying a GDG data set with a relative index produces an error message. The following examples are unsupported specifications:

```
SITE DCBDSN=MYGDG(0)
SITE DCBDSN=MYGDG(-nnn) or
SITE DCBDSN=MYGDG(+nnn)
```

See [“Steps for using a DCBDSN model to create a new data set” on page 84](#) for more information about DCbdsn.

DEBug

Activates or disables general tracing at the FTP server. One or more traces can be activated with a single debug parameter with the following options:

?

Displays the status of the traces.

ACC

The ACC trace shows the details of the login process.

ALL

This parameter is used to set all of the trace points.

Note: Both the FSC and the SOC traces will be set to level 1 when the ALL parameter is processed.

BAS

This parameter is used to set a select group of traces that offer the best overall details without the intense tracing of some of the traces. Specifying this parameter is the same as `SITE DEBUG=(CMD,INT,FSC, SOC)`.

CMD

The CMD trace shows each command and the parsing of the parameters for the command.

FLO

The FLO trace shows the flow of control within FTP. It is useful to show which services of FTP are used for an FTP request.

FSC(*n*)

The FSC trace shows details of the processing of the file services subcommands APPE, STOR, STOU, RETR, DELE, RNFR, and RNTD.

This trace can generate very detailed information and therefore allows you to specify levels of detail for the trace points. The level (*n*) can be a number from 1 to 5.

INT

The INT trace shows the details of the initialization and termination of the FTP session with the server.

JES

The JES trace shows details of the processing for JES requests (that is, requests when `SITE FILETYPE=JES` is in effect).

NONE

This parameter is used to turn off all of the traces.

PAR

The PAR trace shows details of the FTP command parser. It is useful in debugging problems in the handling of the command parameters.

SEC

The SEC trace shows the processing of security functions such as TLS and GSSAPI negotiations.

SOC(*n*)

The SOC trace shows details of the processing during the setup of the interface between the FTP application and the network as well as details of the actual amounts of data that is processed. This trace can generate very detailed information and therefore allows you to specify levels of detail for the trace points.

The level 1 tracing that is specified by entering SOC or SOC(1) is the level normally used unless more data is requested by the TCP/IP service group. The level (*n*) can be a number from 1 to 8.

SQL

The SQL trace shows details of the processing for SQL requests, such as requests when `SITE FILETYPE=SQL` is in effect.

UTL

The UTL trace shows the processing of utility functions such as CD and SITE.

Xyyy

This syntax is used to turn off (reset) a trace that is named by *yyy*. For example, `SITE DEBUG=(XPAR, XACC)` will turn off the PAR and the ACC traces.

Usage notes for the DEBUg parameter:

- The client is allowed to change general tracing at the server if the FTP.DATA file for the server specified `DEBUGONSITE TRUE`.

- The state of the debug traces points is displayed as a response to the SItE subcommand. To see the state without making a change, enter SITE DEBUG=(?).
- The setting of the traces is additive. This is demonstrated by the following example:

```
SITE DEBUG=(BAS)
200-Active traces: CMD INT FSC(1) SOC(1)
200 Site command was accepted
SITE DEBUG=(ACC)
200-Active traces: CMD INT ACC FSC(1) SOC(1)
200 Site command was accepted
```

- To ensure that only the needed traces are active, use the NONE value to clear all traces before setting the requested ones.

```
SITE DEBUG=(?)
EZA1701I >>> SITE DEBUG=(?)
200-Active traces: CMD INT FSC(1) SOC(1)
200 Site command was accepted
SITE DEBUG=(NONE,FSC(2))
EZA1701I >>> SITE DEBUG=(NONE,FSC(2))
200-Active traces: FSC(2)
200 Site command was accepted
```

- For the FSC and SOC trace options, only one level of tracing can be defined at any time. However, when level 2 is defined, levels 1 and 2 are active. When level 3 is defined, levels 1, 2, and 3 are active. This progression also applies to levels 4 and 5.

```
site debug=(fsc(2),soc(1))
>>> SITE debug=(fsc(2),soc(1))
200-Active traces: FSC(2) SOC(1)
200 Site command was accepted
site debug=(fsc(1),soc(2))
>>> SITE debug=(fsc(1),soc(2))
200-Active traces: FSC(1) SOC(2)
200 Site command was accepted
```

Note: The FSC command accepts level values 6–8, but provides only level 5 trace data. Likewise, the SOC trace option accepts level values 4–8, but provides only level 3 trace data.

- See the [Diagnosing FTP server problems with traces](#) section in [z/OS Communications Server: IP Diagnosis Guide](#) for more information about FTP server tracing.

DESt

Specifies the Network Job Entry (NJE) destination to which the files are routed when you enter a PUT command. If specified without a *destination* parameter, the destination resets and files are stored at the host system rather than sent to a remote network.

The SItE DESt subcommand enables you to send data sets (rather than storing them at the server) to other users on machines that are connected on an NJE network.

destination

Specifies the NJE destination to which the files are routed when you enter a PUt command. The value specified for destination can be:

- userID@nodeID
- nodeID.userID
- nodeID
- DestID

The file is sent over the NJE network to the specified destination.

This parameter is ignored if FILEtype=JES is set.

Directory

Specifies the number of directory blocks to be allocated for the directory of a PDS. When specified without the *size*, no directory value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify Directory without a *size* when you are also specifying DATAClass=dataclass and you want the SMS data class to provide the Directory *size*. If Directory=*size* is specified with DATAClass, the value specified by the SIte Directory parameter overrides the DATAClass directory specification.

size

Specifies the number of directory blocks to be allocated for the directory of a PDS. The valid range is 1–16 777 215.

DIRECTORYMode

Specifies that only the data set qualifier immediately below the current directory is treated as an entry in the directory. In directory mode, this data set qualifier is the only one used by the MGet, LS, and DIR subcommands.

DIRECTORYMode has no effect on files residing in a z/OS UNIX file system.

DSNTYPE

Specifies the data set name type for new physical sequential data sets.

SYSTEM

Physical sequential data sets are allocated with the SMS data class value. If no data class is defined, or if the DSNTYPE attribute is not defined, new physical sequential data sets will be allocated with the system default value.

BASIC

Allocates physical sequential data sets as physical sequential basic format data sets.

LARGE

Allocates physical sequential data sets as physical sequential large format data sets.

DSWAITTIME

Specifies the number of minutes FTP waits when trying to access an MVS data set on the FTP server.

minutes

The number of minutes to wait for a local MVS data set to become available. Valid values are in the range 0-14400. If you specify the value 0, the FTP server does not wait to obtain a data set when the data set is being held by another job or process.

Restriction: The FTP server ignores the DSWAITTIME value when processing RENAME FROM (RNFR), RENAME TO (RNT0), DELETE (DELE), and APPEND (for things that are not supported) commands.

DSWAITTIMEREPLY

Specifies the interval for sending a line of the reply 125-Data set access will be retried in 1 minute intervals -- <number> attempts remaining to the client when the FTP server is waiting for access to a data set.

seconds

The number of seconds between reply lines 125-Data set access will be retried in 1 minute intervals -- <number> attempts remaining that the server sends to the client when the FTP server is waiting for access to an MVS data set. The valid range is 15 to 60. The default is 60 seconds.

DUMP

Activates or disables extended tracing at the FTP server.

Note: Extended tracing has the potential to generate a large amount of trace data and should not be set unless requested to debug a specific problem in the code.

One or more traces can be activated with a single dump parameter with the following options:

?

Displays the status of the traces.

n

Specifies the ID number of a specific extended trace point that is to be activated in the FTP code. The number has a range of 1–99.

ALL

This parameter is used to set all of the trace points.

FSC

Activates all of the extended trace points in the file services code. The ID numbers for FSC are 20 to 49.

JES

Activates all of the extended trace points in the JES services code. The ID numbers for JES are 60 to 69.

NONE

This parameter is used to turn off all of the traces.

SOC

Activates all of the extended trace points in the network services code. The ID numbers for SOC are 50 to 59.

SQL

Activates all of the extended trace points in the SQL services code. The ID numbers for SQL are 70 to 79.

Xyyy

This syntax is used to turn off (reset) a trace that is named by yyy. For example, SITE DUMP=(X21,X22,XSQL) will reset the extended trace points 21 and 22 and all of the SQL trace points.

Usage notes for the DUMP parameter:

- The client is allowed to change extended tracing at the server if the FTP.DATA file for the server specified DUMPON SITE TRUE.
- The setting of the traces is additive. This is demonstrated by the following example:

```
SITE DUMP=(NONE,21)
EZA1701I >>> SITE DUMP=(NONE,21)
200-Active dumpIDs: 21
200 Site command was accepted
SITE DUMP=(22)
EZA1701I >>> SITE DUMP=(22)
200-Active dumpIDs: 21 22
200 Site command was accepted
```

- The range of 99 extended trace points is defined to allow easy extension of the trace points by the TCP/IP service team. Additional trace points can be added to the code without any changes to the external mechanism to control the traces.
- See [Diagnosing FTP server problems with traces in z/OS Communications Server: IP Diagnosis Guide](#) for more information about FTP server tracing.

EATTR

Specifies whether newly allocated data sets can have extended attributes and whether new data sets can reside in the EAS of an EAV.

SYSTEM

The data set uses the SMS data class EATTR value. If no SMS data class is defined, or if the data class contains no EATTR specification, the data set is allocated with the system default.

NO

The data set cannot reside in the EAS, and its VTOC entry cannot contain extended attributes.

OPT

The data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

ENCODING

Specifies the kind of encoding that is used for conversions between codepages for data transfers.

See [“Support for SBCS languages” on page 77](#) and [“Support for MBCS languages” on page 81](#) for more information.

SBCS

Single Byte encoding. Code pages are specified using the SBDATACONN statement. This is the default value.

MBCS

Multibyte encoding. Code pages are specified using the MBDATACONN statement.

FIFOIOTIME

Specifies the maximum length of time that the FTP server waits for an I/O operation to a z/OS UNIX named pipe in its z/OS UNIX file system to complete.

Rules:

- When the server sends the contents of a z/OS UNIX named pipe to the client, the FTP server reads from the named pipe one or more times. Each read from the named pipe must complete within the length of time that is specified by the FIFOIOTIME value.
- When you store a file that is received from the client as a z/OS UNIX named pipe, the server writes to the named pipe one or more times. Each write to the named pipe must complete within the length of time that is specified by the FIFOIOTIME value.

seconds

The number of seconds that the FTP server waits for an I/O operation to a z/OS UNIX named pipe to complete. Valid values are in the range 1-86400. The default value is 20.

FIFOOPENTIME

Specifies the length of time that the FTP server waits for an open of a z/OS UNIX named pipe in its z/OS UNIX file system to complete.

seconds

The number of seconds that the FTP server waits for an open of a z/OS UNIX named pipe to complete. Valid values are in the range 1-86400. The default value is 60.

FILEtype

Specifies the file type of the data set.

type

The file type of the data set can be:

Type

Description

SEQ

Sequential or partitioned data sets

SQL

SQL query function

JES

Remote job submission

ISPFSTATS

Allows FTP to create or update ISPF Member statistics when PUt, MPut or APpend subcommands are issued.

JESENTRYLimit

JESENTRYLimit specifies how many entries can be displayed at once using a LIST or NLST command. JESENTRYLIMIT is only valid if JESINTERFACELVEL=2.

number

Fixed number of entries to display. The default for JESENTRYLimit is 200, if not specified in the server FTP.DATA file.

JESGETBYDSN

Specifies that the foreign file that is specified when retrieving a file with FILETYPE JES and JESINTERFACELEVEL 2 is a JES spool data set name to be retrieved for the client.

JESJOBName

Specifies that any command (Get, LIST, DIr, or MGet) should be limited to those jobs, started tasks, APPC/MVS, or TSO output that match the specified value. JESJOBName is only accepted if JESINTERFACELevel is set to 2.

jobname

Specified job name. Can be or can contain a wildcard (* or ?).

loginuserid*

The logged in user ID appended with an asterisk (*). Default value.

Note: JESJOBName matches the first job name that a job is assigned. Jobs that change job names during execution time are matched only by their initial job name.

JESLrecl

Specifies the logical record length (LRecl) for the Job Entry System (JES) internal reader at the foreign host.

length

The logical record length for the JES internal reader at the foreign host. The valid range is 1–254.

Indicates that the logical record length should be taken from the site LRecl parameter setting.

JESOwner

Specifies that any command (Get, LIST, DIr or MGet) should be limited to those jobs, started tasks, APPC/MVS, or TSO output which are owned by the user ID specified. JESOWNER cannot be modified unless JESINTERFACELEVEL is set to 2.

userid

Specified user ID. The *userid* can be or contain a wild card (* or ?).

loginuserid

The logged in user ID. Default value.

JESRecfm

Specifies the record format for the JES internal reader at the foreign host.

F

Fixed record format

V

Variable record format

Indicates that the record format should be taken from the SItE RECfm parameter setting.

JESStatus

Specifies what type of information should be returned on LIST and NLST commands. Acceptable values are INPUT, ACTIVE, OUTPUT or ALL. The default value for JESSTATUS is ALL. JESSTATUS can be modified only if JESINTERFACELEVEL=2.

LISTLEVEL

Specifies the format of the LIST reply.

0

Specifies that PDS, PDSE, and HFS data sets are displayed with a DSORG value of PO.

1

Specifies that PDS data sets are displayed with a DSORG value of PO, PDSE data sets are displayed with a DSORG value of PO-E, and HFS data sets are displayed with a DSORG value of HFS.

2

Specifies the LISTLEVEL 1 options, and also that fewer but wider columns of output are displayed to accommodate larger physical sequential data sets.

LISTSUBdir

Use the LISTSUBdir option to indicate that wildcard searches should apply to the current working directory and should also span one level of its subdirectories. This setting affects processing of the NLST command. The z/OS FTP client sends an NLST command to the server as part of LS*, MDelete *, and MGet * subcommand processing.

Restrictions:

1. The LISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.
2. The FTP client must be communicating with a z/OS V1R7 or later FTP server or an unrecognized parameter response will be received.

Result: If the LISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the client FTP.DATA file, the default is as if the LISTSUBdir option was specified on the SITE subcommand.

LRecl

Specifies the logical record length (LRecl) of a newly allocated data set. When specified without a *length*, no LRecl is used when allocating the data set. The equal sign (=) is optional in this case.

Specify LRecl with no value when you are also specifying *DATAclass=**data_class* and you want the SMS data class to provide the LRecl value, or when you are specifying *DCbdsn=**data_set_name* and you want to use the LRecl from the DCBDSN data set. If *LRecl=length* is specified with either *DATAclass* or *DCbdsn*, the length specified by the Site LRecl parameter overrides the *DATAclass* or *DCbdsn* LRecl.

length

Specifies the logical record length of a newly allocated data set. The valid range is 0–32760. A special value of x (*LRecl=x*) is also supported to indicate that a logical record length can exceed 32760 for variable-length spanned records.

Specifying *LRecl=0* has the same effect as specifying LRecl with no parameters.

MBDATACONN=(*file_system_codepage*,*network_transfer_codepage*)

Specifies the codepages for the file system and for the network transfer used when the server does data conversion during a data transfer. This parameter affects the conversion of multibyte character set (MBCS) data (including support for DBCS code pages) and is used when the *ENCODING=MBCS* is also specified.

See “Support for MBCS languages” on page 81 for more information.

file_system_codepage

Specifies the name of the file system codepage.

network_transfer_codepage

Specifies the name of the network transfer codepage.

MBREQUIRELASTEOL

Specifies that the FTP server will report an error when a multibyte file or data set is received from the network with no EOL sequence in the last record received. FTP will abort the file transfer.

MBSENDEOL

Specifies which end-of-line sequence to use when *ENCODING* is MBCS and data is being sent to the client and translated to ASCII.

CRLF

Append both carriage return (X'0D') and line feed (X'0A') end-of-line sequences to each line of translated text. This is the default and the standard sequence defined by RFC 959.

CR

Append only a carriage return (X'0D') end-of-line sequence to each line of translated text.

LF

Append only a line feed (X'0A') end-of-line sequence to each line of translated text.

NONE

Do not append an end-of-line sequence to the line of translated text.

Requirements:

- Most FTP clients support only the CRLF value for incoming ASCII data. Do not specify another value for MBSSENDEOL unless you have verified that the client is expecting the end-of-line sequence that you specify.
- Do not use an end-of-line sequence other than CRLF if the client is a z/OS FTP client. The z/OS FTP client supports only the CRLF value for incoming type ASCII data.
- Do not attempt to stream mode restart a multibyte file retrieve that originated while the FTP server MBSSENDEOL value was not CRLF.

MGmtclass

Specifies the SMS management class as defined by your organization for the target host. Specifying MGmtclass with no *mgmtclass* cancels the management class specification. The equal sign (=) is optional in this case.

mgmtclass

Specifies the SMS management class as defined by your organization for the target host. If the mgmtclass specified has a setting for RETpd, the value specified by the mgmtclass can override the setting of the RETpd site parameter, the RETpd value of a model data set if the DCbdsn parameter is specified, and the RETpd value defined in an SMS data class if DATAclass is specified. See [“Specifying values for new data sets” on page 82](#) for more information about specifying attributes when allocating new data sets.

MIGratevol

Specifies the volume ID for migrated data sets if they do not use IBM storage management subsystems. If you do not specify MIGratevol, the default *volume_serial* is MIGRAT.

volume_ID

The volume ID for migrated data.

NOASAtans

Treats ASA file transfers as regular file transfers; that is, the ASA characters are treated as part of the data and are not converted to print control characters.

NOAUTOMount

Prevents automatic mounting of volumes for data sets on volumes that are not mounted.

NOAUTOREcall

Prevents automatic recall of migrated data sets.

Note: A migrated data set can be deleted even though NOAUTOREcall is specified, because migrated data sets are not recalled for deletion.

NODBSUB

Specifies that substitution is not allowed for data bytes that cannot be translated in a double-byte character translation. This causes a data transfer failure if a character cannot be translated during the transfer. This is the default.

NOISPFSTATS

Does not allow FTP to create or update ISPF member statistics when PUt, MPut, or APpend subcommands are issued.

NOJESGETBYDSN

Specifies that the foreign file that is specified when retrieving a file with FILETYPE=JES is a file on the MVS system that is to be submitted to JES as a batch job.

NOLISTSUBdir

Use the NOLISTSUBdir option to indicate that wildcard searches should apply only to the current working directory. This setting affects processing of the NLST command. The z/OS FTP client sends an NLST command to the server as part of LS*, MDelete *, and MGet * subcommand processing.

Restrictions:

1. The NOLISTSUBdir option applies to z/OS UNIX file operations only; MVS data set operations are not affected.
2. The FTP client must be communicating with a z/OS V1R7 or later FTP server or an *unrecognized parameter* response will be received.

Result: If the NOLISTSUBdir option is not specified on the SITE subcommand and the LISTSUBDIR statement is not specified in the client FTP.DATA file, the default is as if the LISTSUBdir option was specified on the SITE subcommand.

NOMBREQUIRELASTEOL

Specifies that the FTP server will not report an error when a multibyte file or data set is received from the network with no EOL sequence in the last record received. FTP will report the file transfer as completed.

NOQUOTEsoverride

Treats a single quotation mark appearing at the beginning of the file name, as well as all other single quotation marks contained in the file name, as part of the actual file name. The entire file name, including the leading single quotation mark, is appended to the current working directory.

NORDW

Specifies that Record Descriptor Words (RDWs) are not treated as if they were part of the record and are discarded during FTP transmission of variable format data sets.

NOREMOVEINBEOF

Specifies that the UNIX end-of-file (EOF) byte (X'1A') is not removed on inbound ASCII transfers before the data is stored. See [z/OS Communications Server: IP Configuration Reference](#) for more information.

NORESTPUT

Specifies that the FTP server does not support checkpoint or restart processing when it is receiving data.

NOSBSUB

Specifies that substitution is not allowed for data bytes that cannot be translated in a single-byte character translation. This causes a data transfer failure if a character cannot be translated during the transfer.

NOSPRead

Specifies that the output is in report format rather than spreadsheet format when the file type is SQL.

NOTAPEREADSTREAM

Specifies that a common read path is used for retrieving tape data sets from the server. This is the default value.

NOTRAILINGblanks

Specifies that the FTP server does not preserve the trailing blanks that are in a fixed format data set when the data is sent to the foreign host.

NOTRUNcate

Specifies that truncation is not permitted. The FTP server will set an error and fail file transfer if a record that is longer than the LRECL of the new file is detected.

Note: If WRAPRECORD is set then the data is wrapped, not truncated, no error will be set and the file transfer will continue.

NOUCSSUB

In UCS-2-to-EBCDIC conversion, the data transfer is terminated if any UCS-2 character cannot be converted into the EBCDIC code set.

NOUCSTRUNC

In UCS-2-to-EBCDIC conversion, truncation of EBCDIC data is not allowed. The data transfer is aborted if the logical record length of the receiving data set is too small to contain the data after conversion to EBCDIC.

Note: The setting of the CONDdisp parameter determines what happens to the target data set if the transfer is aborted.

NOWRAPrecord

Specifies that data is truncated if no new line character is encountered before the logical record length of the receiving file is reached.

Note: If NOWRAPrecord and NOTRUNcate are set, then an error will be set and the file transfer will fail.

NOWRTAPEFastio

Specifies that ASCII Stream data that is being written to tape must be written using the Language Environment runtime library.

PDSTYPE

Specifies whether the FTP server creates MVS directories as partitioned data sets or as partitioned data sets extended.

When specified without a value, FTP will not specify to z/OS whether to allocate a new MVS directory as a PDS or a PDSE. When specified without a value, the equal sign (=) is optional.

PDS

Allocate directories as partitioned data sets.

PDSE

Allocate directories as partitioned data sets extended.

PRImary

Specifies the number of tracks, blocks, or cylinders for primary allocation. When specified without an *amount*, no primary value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify PRImary with no value when you are also specifying DATAclass=*data_class* and you want the SMS data class to provide the PRImary *amount*.

To allow the SMS data class to determine the space allocation, both PRImary and SECondary must be specified with no value. The tracks, blocks, and cylinders setting is ignored in this case. If PRImary=*amount* is specified with DATAclass, the value specified by the SItE PRImary parameter overrides the DATAclass space allocation.

amount

Specifies the number of tracks, blocks, or cylinders for primary allocation. For allocating partitioned data sets, this is the amount of space that is allocated for the primary extent.

For allocating sequential data sets this is the maximum amount of space that is allocated for the primary extent. If a smaller amount of space is needed to hold the data being transferred, only the amount actually needed to hold the data is allocated. The valid range is 1 - 16 777 215.

Qdisk

Used to display statistics about available space on a volume. If the Qdisk parameter is entered without a specific *volume_serial*, statistics about available space are displayed for each volume that is defined with "Use Attribute=storage."

volume_serial

Displays statistics about available space on a specific volume.

QUOTESoverride

Specifies that single quotation marks at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory. This is the way single quotation marks are used in all previous MVS FTP servers and is the default. Any single quotation mark inside the beginning and ending quotation marks is treated as part of the file name.

QUOTESoverride indicates the usage of single quotation mark appearing at the beginning of, or surrounding, a file name. The setting of this keyword affects all FTP subcommands that have a path name as a parameter except keywords on the SITE subcommand.

RDW

Specifies that Record Descriptor Words (RDWs) are treated as if they were part of the record and are not discarded during FTP transmission of variable format data sets in stream mode.

Note: RDW information is stored in binary format. Transfer files in binary mode to avoid translation problems that can occur if you transfer this binary field in EBCDIC or ASCII mode.

READTAPEFormat

Used to provide information about an input data set on tape. If specified without the *tape_format* (which is the default), processing of input tapes does not take advantage of the record format information prior to open. The equal sign (=) is optional in this case.

READTAPEFormat has no effect on, and is not affected by DATAClass, DCbdsn, JESLrecl, JESRecfm, LRecl, RECFM, or any other parameters associated with creating a data set.

tape_format

Specifies the format of the records on the input tape data set. Valid formats are:

F

Fixed record length

V

Variable record length

S

Spanned records

X

Logical record length is undefined (Lrecl X)

blank

Unspecified (displayed as *U* in messages and reply)

The formats are mutually exclusive. Spanned implies variable and Lrecl X implies spanned. If specified, the *tape_format* value must be the most inclusive identifier in the list that matches the tape. If it is not the most inclusive identifier, an error message is issued. For example, if the *tape_format* value is **S** (spanned) and the tape contains records with undefined length (Lrecl X), the request fails. An unspecified format avoids this type of error. However, the following should be considered:

- Specify a value for READTAPEFormat in all the following cases. Failure to specify a format will likely cause errors in processing the tape.
 - The record length is undefined (Lrecl X).
 - The records are spanned (Recfm is VBS, VS).
 - The records are variable (Recfm is V, VB, VBA) and RDW is specified.
- Specify a value for READTAPEFormat for all input tapes with one of the listed formats to ensure best results.

RECFM

Specifies the record format of a data set. When specified without the *format*, no record format is used when allocating the data set. The equal sign (=) is optional in this case.

Specify RECfm with no value when you are also specifying DATAclass=*data_class* and you want the SMS data class to provide the RECfm *format*, or when you are specifying DCbdsn=*data_set_name* and you want to use the record format from the DCBDSN data set.

If RECfm=*format* is specified with either DATAclass or DCbdsn, the value specified by the SItE RECfm parameter overrides the DATAclass or DCbdsn record format.

format

Specifies the record format of a data set. Valid record formats are: F, FA, FB, FBA, FBM, FBS, FBSA, FBSM, FM, FS, FSA, FSM, U, UA, UM, V, VA, VB, VBA, VBM, VBSA, VBSM, VBS, VM, VS, VSA, and VSM. The characters used to specify these record formats have the following meanings:

| Code | Description |
|-------------|---|
| F | Fixed record length |
| V | Variable record length |
| U | Undefined record length |
| B | Blocked records |
| S | Spanned records (if variable) / standard records (if fixed) |
| A | Records contain ISO/ANSI control characters |
| M | Records contain machine code control characters |

REMOVEINBOF

Specifies that the UNIX end-of-file (EOF) byte (X'1A') is removed before the data is stored on inbound ASCII transfers. See [z/OS Communications Server: IP Configuration Reference](#) for more information.

RESTPUT

Specifies that the FTP server supports checkpoint or restart processing when it is receiving data.

RETpd

Specifies the number of days that a newly allocated data set should be retained. When specified without the number of *days*, a retention period will not be specified when allocating new data sets. The equal sign (=) is optional in this case.

Specify RETpd with no value when you are also specifying DATAclass=*data_class* or MGmtclass=*mgmtclass* and you want SMS to provide the RETpd value, or when you are specifying DCbdsn=*data_set_name* and you want to use the RETpd from the DCBDSN data set. If more than one of the SItE parameters (RETpd, MGmtclass, DATAclass, or DCbdsn) are specified, the order of precedence (highest to lowest) is:

1. MGmtclass
2. RETpd
3. DCbdsn
4. DATAclass

If a retention period is associated with an SMS management or data class, or with a model DCBDSN data set, the value of the retention period can be overridden to another retention period, but it cannot be overridden to have no retention period specified for the newly created data sets.

days

Specifies the number of days that a newly allocated data set should be retained. The valid range is 0–9999. A value of 0 indicates a retention period of 0 days so that the data set expires the same day it was created.

Note: An attempt to either append or replace an existing data set with a retention period requires operator interaction to take place for permission to alter the data set. This is normal MVS behavior.

SBDataconn

Specifies the conversions between file system and network code pages to be used for data transfers. Valid subcommands are:

```
SITE SBDataconn=dsname
SITE SBDataconn=(file_system_cp,network_transfer_cp)
SITE SBDataconn=FTP_STANDARD_TABLE
SITE SBDataconn=*
SITE SBDataconn=
SITE SBDataconn
```

The following forms of specifying SBDataconn are equivalent to specifying SBDataconn=*

- SBDataconn
- SBDataconn=

See [“Support for SBCS languages”](#) on page 77 for more information.

dsname

Specifies the fully qualified name of an MVS data set or z/OS UNIX file that contains the EBCDIC-to-ASCII and ASCII-to-EBCDIC translate tables generated by the CONVXLAT utility.

Notes:

1. The name must *not* be enclosed in quotation marks. If quotation marks appear, they are treated as part of the name. (QUOTESoverride is ignored.)
2. The z/OS UNIX file system name is case sensitive. The MVS name is not case sensitive.
3. The name cannot begin with a left parenthesis [(.]
4. The SBDataconn keyword must be the only keyword or the last keyword on a SItE subcommand.
5. The translate tables being used for the data connection can also be changed by a SItE XLate subcommand.
6. SItE XLate and SItE SBDataconn are mutually exclusive.

file_system_cp

Specifies the name of a code page recognized by iconv. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

network_transfer_cp

Specifies the name of a code page recognized by iconv. For a list of code pages supported by iconv, see code set converters information in the [z/OS XL C/C++ Programming Guide](#).

FTP_STANDARD_TABLE

Specifies that the FTP internal tables, which are the same as the tables that are shipped in TCPXLBIN(STANDARD), are to be used on the data connection.

Specifies the translate tables set up at initialization for the data connection must be used.

SBSSEDEOL

Specifies which end-of-line sequence to use when ENCODING is SBCS, the data type is ASCII, and data is being sent to the client.

CRLF

Append both carriage return (X'0D') and line feed (X'0A') end-of-line sequences to each line of translated text. This is the default and the standard sequence defined by RFC 959. The z/OS server can receive ASCII data in this format only.

CR

Append only a carriage return (X'0D') end-of-line sequence to each line of translated text.

LF

Append only a line feed (X'0A') end-of-line sequence to each line of translated text.

NONE

Do not append an end-of-line sequence to the line of translated text.

Tips:

1. The SIZE command is disabled if you configure a SBSENDEOL value other than CRLF.
2. The REST command in Mode Stream is disabled if you configure a SBSENDEOL value other than CRLF. A mode block REST command is not affected by the SBSENDEOL setting.
3. SIZE and REST commands are sent by clients as part of a stream mode restart of an interrupted file transfer. Because these commands are disabled by changing the SBSENDEOL value from the RFC 959 standard, stream mode restarts are effectively disabled. Block mode restart is not affected by the SBSENDEOL setting.

Rules:

1. Most clients support only the CRLF value for incoming type ASCII data. Do not specify another value for SBSENDEOL unless you have verified that the client is expecting the EOL sequence that you specify.
2. Do not use an EOL sequence value other than CRLF if the client is a z/OS FTP client. The z/OS FTP client supports only the CRLF value for incoming ASCII data.

SBSUB

Specifies that substitution is allowed for data bytes that cannot be translated in a single byte character translation. The substitution character is specified by the SBSUBCHAR parameter.

SBSUBCHAR *nn*

Specifies the value that is used for substitution when SBSUB is also specified. The value is one of the following:

SPACE

When the target code set is ASCII, replace untranslatable characters with X'20' during SBCS data transfers. When the target code set is EBCDIC, replace untranslatable characters with X'40' during SBCS data transfers.

nn

Replace untranslatable characters with *nn* during SBCS data transfers where *nn* is a hexadecimal value from 00 to FF.

SECondary

Specifies the number of tracks, blocks, or cylinders for secondary allocation. When specified without the *amount* for the C server, no secondary value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify SECondary with no value when you are also specifying DATAClass=*dataclass* and you want the SMS data class to provide the SECondary value. To allow the SMS data class to determine the space allocation, both PRImary and SECondary must be specified with no value. The tracks, blocks, or cylinders setting is ignored in this case. If SECondary=*amount* is specified with DATAClass, the value specified by the SIte SECondary parameter overrides the DATAClass space allocation.

amount

Specifies the amount of tracks, blocks, or cylinders for secondary allocation. The valid range is 0–16 777 215.

SPRead

Specifies that the output is in spreadsheet format when the file type is SQL.

SQLCol

Specifies the column headings of the SQL output file.

any

The label of the Db2 SQL table column heading is the first choice for column heading, but if there is no label, the name becomes the column heading.

labels

Labels are the Db2 SQL table column headings. If any of the columns do not have labels, FTP supplies a column heading in the form of COLnnn.

names

Uses the names of the Db2 SQL table column headings. The labels are ignored.

STORclass

Specifies the SMS storage class for the target host, as defined by your organization. Cancels the storage class specification when specified without a *storage_class* parameter value. The equal sign (=) is optional in this case.

See [“Specifying values for new data sets” on page 82](#) for more information about specifying attributes when allocating new data sets.

storage_class

Specifies the SMS storage class as defined by your organization for the target host.

When an SMS storage class is in use, any of the attributes specified there can be overridden by the user by a different specification. To avoid overriding the setting in the SMS storage class, specify BLKSize, LRecl, PDSTYPE, PRImary, RECfm, SECondary, UCOUNT, Unit, VCOUNT, or VOLUME with no associated value. This removes any value specified on a prior SITE command or in FTP.DATA, and the affected attributes are not included on the allocation. To override a setting in the SMS storage class, specify the wanted value with the appropriate keyword.

SUBSYS

Specifies the name of the subsystem that is to be used when allocating data sets. If you specify the SUBSYS parameter without a subsystem name, the subsystem support is disabled.

Tip: You can use the SUBSYS parameter to transfer files to BatchPipes. See [“SUBSYS: Writing to BatchPipes” on page 135](#).

Restrictions: The following restrictions apply when a SUBSYS value is specified:

- APPE and REST commands are not supported.
- Only binary (type I) file transfer is supported.
- Only FILETYPE SEQ is supported.
- Checkpointing and file transfer restart are not supported. Checkpointing is described in [“Restarting a failed data transfer” on page 103](#).
- Do not use with SMS-managed data sets (data sets with an assigned storage class).
- Only RECFM values F, FB, V, and VB are supported.

TAPEREADSTREAM

Specifies that a more efficient read path (read as stream) is used for retrieving tape data sets from the server.

Restriction: If a SITE TAPEREADSTREAM subcommand is issued:

- You cannot retrieve American Standards Association (ASA) tape data sets. The server responds with an error reply if you attempt to retrieve an ASA tape data set.
- You cannot retrieve fixed format tape data sets when TRAILINGBLANKS TRUE is configured. The server responds with an error reply if you attempt to retrieve a fixed format tape data set when TRAILINGBLANKS TRUE is configured.
- If the tape data set contains <NL> characters that require translation, the data set format is incorrect.

TRacks

Specifies that primary and secondary space allocations are in tracks. If both PRImary and SECondary values are unspecified, and an SMS data class value has been specified, the space allocation is determined by the SMS data class value and the TRacks parameter value is ignored.

TRAILingblanks

Specifies that the FTP server preserves the trailing blanks that are in a fixed format data set when the data is retrieved from a foreign host.

TRUNcate

Specifies that truncation is permitted. The FTP server does not set an error when a truncated record is detected and file transfer continues.

UCOUNT

Specifies how many devices to allocate concurrently to support the allocation request.

unit_count

Specifies number of devices to allocate. Valid values are in the range 1-59. When specified without a value, the FTP server does not specify a unit count when allocating data sets.

P

Parallel mount request.

Guideline: The UCOUNT statement is not meant to be used with an SMS storage class. Any UCOUNT value you specify overrides whatever is specified for the SMS managed dataclass that being used.

UCSHOSTCS

Specifies the EBCDIC code set to be used when converting to and from UCS-2. If you do not specify a *code_set* value, the current code set is used.

code_set

Name of the EBCDIC code set to be used when converting to and from UCS-2.

UCSSUB

In UCS-2 to EBCDIC conversion, the EBCDIC substitution character is used to replace any UCS-2 character that cannot successfully be converted. Data transfer continues.

UCSTRUNC

In UCS-2 to EBCDIC conversion, truncation of EBCDIC data is allowed. The data transfer continues even if EBCDIC data is truncated.

Note: If the EBCDIC data contains any double-byte data, truncation might not honor character boundaries and EBCDIC records might not end in Shift-in state.

UMask

Defines the file mode creation mask. The file mode creation mask defines which permission bits are *not* to be set on when a file is created. When a file is created, the permission bits requested by the file creation are compared to the file mode creation mask, and any bits requested by the file creation which are disallowed by the file mode creation mask are turned off.

The format of the UMask keyword is UMASK *ooo*.

When a file is created, the specified permission bits for the file are 666 (-rw-rw-rw-). If the file mode creation mask is 027, the requested permissions and the file mode creation mask are compared:

```
110110110   - 666
000010111   - 027
-----
11010000    - 640
```

The actual permission bits set for the file when it is created is 640 (-rw-r-----).

Notes:

1. The default value for UMask is 027.
2. You cannot use FTP to create z/OS UNIX files with execute permissions. If you require execute permissions, use the SITE CHMod command to change permissions after the file has been created.

UNICODEFILESYSTEMBOM

Specifies whether the FTP server stores incoming Unicode files with a byte order mark (BOM).

Restriction: The only Unicode encoding formats supported for file storage by z/OS FTP are UTF-8 and UTF-16. Files are always stored in big endian format.

Result: The byte order mark (BOM) stored with the file is determined by the encoding used to store the file rather than by the format of the BOM sent with the file.

ASIS

Store incoming Unicode files with a byte order mark only if the file was sent with a byte order mark.

ALWAYS

Store incoming Unicode files with a byte order mark regardless of whether the file was sent with a byte order mark.

NEVER

Store incoming Unicode files without a byte order mark regardless of whether the file was sent with a byte order mark.

Results:

- The Unicode byte order mark, U+FEFF, can also be interpreted as a zero-width nonbreaking space character. z/OS FTP considers only the first character of the data that is received from the client as a possible byte order mark (BOM). No other instance of the BOM sequence in the inbound data is affected by this setting.
- When you are appending to a nonexistent file, the FTP server respects the UNICODFILESYSTEMBOM setting. However, when you are appending to an existing file, the FTP server always strips a leading BOM from the incoming file. This prevents a superfluous BOM from being inserted in the server file.
- When the server file is a z/OS UNIX named pipe, incoming data is always appended to any existing data that is in the named pipe. If you code UNICODFILESYSTEMBOM = ASIS or ALWAYS and the named pipe contains data, the server appends a BOM byte sequence to existing data in cases in which it would add a BOM at the beginning of a regular file. The BOM byte sequence is interpreted as a zero-width nonbreaking space character when it does not start the file or data stream. You must take this into consideration when you configure UNICODFILESYSTEMBOM. See [“Using z/OS UNIX System Services named pipes” on page 104](#) for more information.

Unit

Specifies the unit type for allocation of new data sets.

unit_type

The unit type (for example, 3380) for the allocation of new data sets on direct access devices. If a *unit_type* value is not specified, the unit type used for allocation is restored to the system default.

UNIXFILETYPE

Specifies whether the server treats files in its z/OS UNIX file system as regular files or as named pipes.

FILE

Treat files in the z/OS UNIX file system as regular files. This is the default.

FIFO

Treat files in the z/OS UNIX file system as named pipes.

See [“Using z/OS UNIX System Services named pipes” on page 104](#) for information about transferring data into and from UNIX named pipes.

VCOUNT

Specifies the number of tape data set volumes that an allocated data set can span. When this parameter is specified without a *volume_count* value, the FTP server uses the volume count 50 when it allocates tape data sets.

volume_count

Valid values are in the range 1-255.

VOLUME

Specifies the volume serial number for allocation of new data sets.

volume_serial

The serial number of the volume to use for allocation.

volume_serial_list

A list of one or more volume serial numbers for allocation. Delimit each *volume_serial* from the prior one with a comma.

If a VOLume value is specified without a *volume_serial_list* or *volume_serial* parameter, no volumes are specified by the FTP server during the allocation of a new data set, and the installation default is used.

The MVS FTP server identifies multiple commands issued with a single-site command by the white space. For example (note the white space in the two commands): `site vol=fffff` is a single-site command; however, `site vol = fffff` is treated by the server as three different commands.

WRAPrecord

Specifies that data is wrapped to the next record if no new line character is encountered before the logical record length of the receiving file is reached.

WRTAPEfastio

Specifies that ASCII Stream data that is being written to tape is allowed to be written using BSAM I/O.

XLate

Specifies the wanted translate table to be used for the data connection. Valid subcommands are:

```
SITE XLate=name
SITE XLate=*
```

name

Specifies the name that corresponds to the wanted translate table data set. The corresponding data set name is *hlq.name.TCPXLBIN* unless environment variable `_FTPXLATE_name = dsn` was defined for the FTP server to override the data set name. In that case, *dsn* is the data set used.

Indicates that the translate tables set up at initialization for the data connection are to be used.

Notes:

1. The translate tables being used for the data connection can also be changed by the `SBDatconn` parameter.
2. Site `XLate` and Site `SBDatconn` are mutually exclusive.

Guideline: If you want to store files on the remote host as MVS data sets, use the `SItE` subcommand to send data set allocation attributes to the host.

Tips:

- Use the `HElP` subcommand with the `SERVER SItE` parameters to display information about the `SItE` parameters that are supported by the server you are logged in to.
- You can specify `SItE` subcommand parameters that are not described in this topic. The z/OS FTP client sends all parameters to the remote host for processing. This is useful when the server on the remote host is not a z/OS FTP server. Below is an example:

```
site myUniqueParameter=12
```

- You can send the parameters described in this topic to the FTP server using any FTP client. If the FTP client does not support the `SItE` subcommand, use the `QUOtE` subcommand to send the information to the server. Below is an example:

```
QUOTE SITE EATTR=OPT
```

- You can specify more than one parameter with the `SItE` subcommand. Delimit each parameter with a blank space.
- If you are sending MVS data sets to the remote host, use the `SENDSite` subcommand to toggle the automatic sending of `SItE` commands to the FTP server.

Results:

- The FTP server on the remote host might not implement all the parameters described in this section. This happens often when the FTP server is not a z/OS FTP server.

- The site-dependent information you send with the SItE subcommand remains active until you issue a new SItE subcommand. The new SItE subcommand adds to or changes the attributes established by previous SItE subcommands.
- If you specify one or more incorrect parameters with the SItE subcommand while you are logged in to the z/OS FTP server, an error message that specifies the incorrect parameter is displayed. The z/OS FTP server sets all correct parameters, regardless of any incorrect parameters that were specified.

Related topics:

- For more information about the HElp subcommand, see [“HElp and ? subcommands—Display help information”](#) on page 171.
- To check the effect of the SItE subcommand on the attributes at the foreign host, see [“STAtus subcommand—Retrieve status information from a remote host”](#) on page 278.
- See [“SENDSite subcommand—Toggle the sending of site information”](#) on page 246 for more information about the SENDSite subcommand.
- See [“QUOte subcommand—Send an uninterpreted string of data”](#) on page 240 for more information about the QUOte subcommand.
- See the [z/OS MVS JCL Reference](#) for more information about some of the SItE and LOCSItE parameters.

SJiskANJI subcommand—Change the data transfer type to SJISKANJI

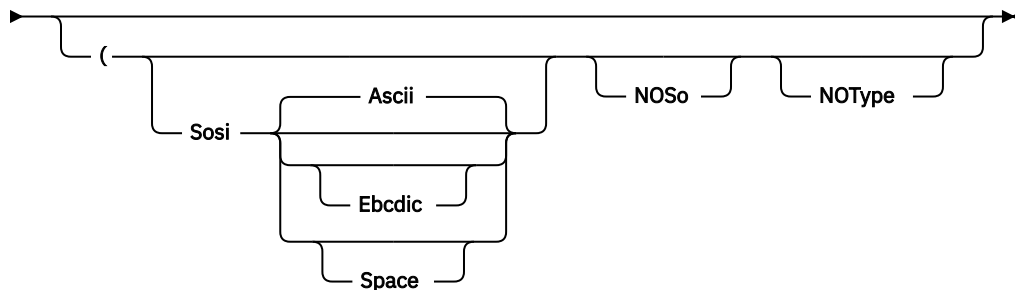
Purpose

Use the SJiskANJI subcommand to change the data transfer type to SJISKANJI.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SItE/LOCSItE SBDataconn or SItE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format

➤ SJiskANJI ➡



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — Ascii, Ebcdic or Space. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space

When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when translation is to be done by the FTP client only.

Examples

To cause the FTP client to change its transfer type to Shift JIS kanji, without sending a TType command to the FTP server, use:

```
SJISKANJI (NOTYPE
```

The server in this example should be set to the ASCII transfer type before the (NOTYPE subcommand is issued.

Usage

- The SJiskANJI client subcommand is equivalent to the TYPE B 1 server command.
- The SJiskANJI (Sosi or SJiskANJI (Sosi ASCII client subcommands are equivalent to the TYPE B 1 S A server command.
- The SJiskANJI (Sosi Ebcdic client subcommand is equivalent to the TYPE B 1 S E server command.
- The SJiskANJI (Sosi SPACE client subcommand is equivalent to the TYPE B 1 S S server command.
- The SJiskANJI (NOSO client subcommand is equivalent to the TYPE B 1 N server command.

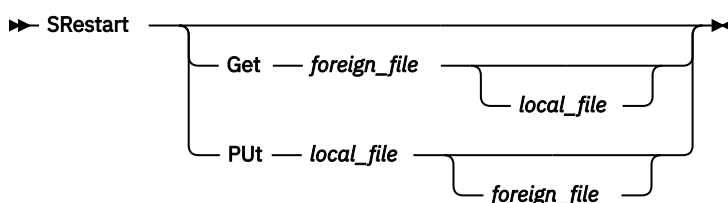
Context

See [“FTP with traditional DBCS support” on page 78](#) and [“Support for MBCS languages” on page 81](#) for more information.

SRestart subcommand—Restart a stream data transfer

Purpose

Use the SRestart subcommand to restart an interrupted stream mode data transfer.

Format

Parameters

Get

Resume a subcommand.

foreign_file

Specifies the name of the file to be retrieved from the remote host.

local_file

Specifies the name of the local file created as a result of the Get subcommand.

Enter the same names for *foreign_file* and *local_file* as you used in the interrupted subcommand.

You can override the usage of the current local working directory in the local file name by specifying *local_file* as a complete data set name enclosed in single quotation marks (''). If *local_file* is not specified, the *local_file* name is the same as the *foreign_file* name.

PUt

Resume a put subcommand.

local_file

Specifies the name of the file on your local host being sent to the remote host.

foreign_file

Specifies the name that the delivered file is given on the remote host. If the *foreign_file* name is not specified, the *foreign_file* name is the same as the *local_file* name.

Enter the same names for *foreign_file* and *local_file* as you used in the interrupted put subcommand.

You can override the usage of the current local working directory in the local file name by specifying *local_file* as a complete data set name enclosed in single quotation marks (''). If *local_file* is not specified, the *local_file* name is the same as the *foreign_file* name.

Usage

- Do not use SRestart to resume an interrupted file transfer unless you can create the start options again, FTP.DATA statements, SItE, and LOCSItE options in effect at the time of the failed file transfer. Unpredictable results can occur if the file transfer environment cannot be created again. If you cannot create the file transfer environment again, issue Get or PUt to transfer the file again.
- Not all file transfers can be resumed in STREAm mode. Observe these restrictions during the original file transfer as well as the SRestart transfer.
 - Mode must be STREAm.
 - Filetype must be SEQ.
 - Unixfiletype must be FILE.
 - Structure must be FILE.
 - SUnique option must be disabled.
 - Data type must be ASCII, EBCDIC, or Image.
 - SITE and LOC SITE ENcoding must be SBCS (SRestart does not support DBCS or MBCS ENcoding).
 - The session cannot be protected with an active security mechanism (such as TLS or Kerberos).
 - For SRestart PUt, LOC SITE SBSENDEOL=CRLF must be set. If the original transfer used a setting other than CRLF, the transfer cannot be restarted. Using a setting other than CRLF will cause the SRestart to fail. Setting SBSENDEOL to CRLF when the original transfer used a different setting will corrupt the remote file.
 - For SRestart Get, SITE SBSENDEOL=CRLF must be set. If the original transfer used a setting other than CRLF, the transfer cannot be restarted. Using a setting other than CRLF will cause the SRestart to fail. Setting SBSENDEOL to CRLF when the original transfer used a different setting will corrupt the local file.
 - The local file must reside in the z/OS UNIX file system as a regular file.

- For SRestart PUT, a server that is z/OS Communications Server V1R2 or later rejects the restart if the *foreign_file* value is not a z/OS UNIX file system regular file.
- For SRestart PUT, if the server cannot calculate an appropriate point to restart the file transfer, SRestart will fail.
- For SRestart Get, if the client cannot calculate an appropriate point to restart the file transfer, SRestart will fail.
- If SRestart fails, use Get or PUT and restart the file transfer.

Context

- See [“SUnique subcommand—Changes the storage method ” on page 286](#) for information about changing the storage method on the remote host.
- See [“STRucture subcommand—Set the file structure ” on page 286](#) for more information.
- See [“LOCSite subcommand—Specify site information to the local host” on page 179](#) for a description of the SBSENDEOL and ENcoding parameters.

STatus subcommand—Retrieve status information from a remote host

Purpose

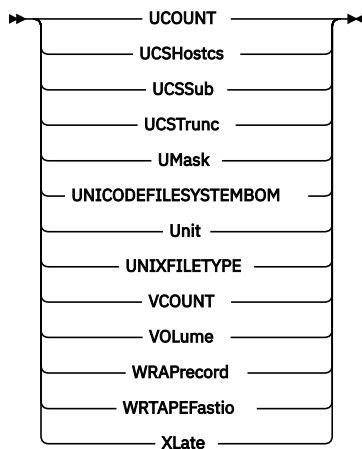
Use the STatus subcommand to retrieve current configuration information from the FTP server. This information includes the current settings of the configuration variables, which can be initialized in the FTP.DATA data set or changed using various FTP subcommands. For information about the parameters of the FTP.DATA data set, see the [z/OS Communications Server: IP Configuration Reference](#).

Format

```
➔ STatus name (option)
```

options

| |
|------------------|
| ASAttrans |
| AUTOMount |
| AUTORecall |
| BLKsize |
| Blocks |
| BLOCKSize |
| BUfno |
| CHKptint |
| CONDdisp |
| Cylinders |
| DATAclass |
| DATAKEEPLIVE |
| DATASetmode |
| DB2 |
| DBSUB |
| DCbdsn |
| DESt |
| Directory |
| DIRECTORYMode |
| DSNTYPE |
| DSWAITTIME |
| EATTR |
| ENcoding |
| FIFOIOTIME |
| FIFOOPENINGTIME |
| FILEtype |
| FTpkeepalive |
| INactivetime |
| ISPFStats |
| JESENTRYLimit |
| JESGETBYDSN |
| JESJOBName |
| JESLrecl |
| JESOwner |
| JESRecfm |
| JESStatus |
| LISTLEVel |
| LISTSUBdir |
| LRecl |
| MBDATACONN |
| MBREQUIRELASTEOL |
| MBSSENDEOL |
| MGmtclass |
| MIGratevol |
| PDSTYPE |
| PRImary |
| QUotesoverride |
| RDw |
| READTAPEFormat |
| RECFm |
| RETPd |
| SBDatconn |
| SBSSENDEOL |
| SBSUB |
| SBSUBChar |
| SEConary |
| SPRead |
| SQLCol |
| STORclass |
| TLRFLEVEL |
| TRacks |
| TRAILingblanks |
| TRUNcate |



Parameters

ASAtans

Indicates that the FTP server interprets characters in the first column of ASA files being transferred as print control characters.

AUTOMount

Indicates automatic mounting of volumes for data sets that are on unmounted volumes.

AUTOREcall

Indicates automatic recall of migrated data sets.

BLOCKS

Indicates that primary and secondary space allocations are in blocks.

BLOCKSize

Indicates the block size of a newly allocated data set.

BUfno

Indicates the number of access method buffers that are to be used when data is read from or written to a data set.

CHKptint

Indicates the checkpoint interval for the sending site in a file transfer request.

CONDdisp

Indicates the disposition of the data set if a retrieve operation for a new data set ends before all of the data is written.

CYLinders

Indicates that primary and secondary space allocations are in cylinders.

DATAclass

Indicates the SMS data class.

DATAKEEPALIVE

Indicates the number of seconds that TCP/IP waits while the data connection is inactive before sending a keepalive packet to the FTP client. The value 0 indicates that the DATAKEEPALIVE timer is disabled for this session. For active mode data connections, the keepalive timer that is configured in PROFILE.TCPIP controls how often keepalive packets flow on the data connection. For passive mode data connections, FTP suppresses the PROFILE.TCPIP keepalive timer.

DATASetmode

Indicates whether DATASetmode or DIRECTORYMode is in effect.

DB2

Indicates the Db2 subsystem name.

DBSUB

Indicates whether substitution is allowed for data bytes that cannot be translated in a double-byte character translation.

DCbdsn

Indicates the name of the MVS data set to be used as a model for allocating new data sets.

DESt

Indicates the Network Job Entry (NJE) destination to which the files are routed when you enter a PUT command.

Directory

Indicates the number of directory blocks to be allocated for the directory of a PDS.

DIRECTORYMode

Indicates whether DATASetmode or DIRECTORYMode is in effect.

DSNTYPE

Indicates the data set name type for new physical sequential data sets.

SYSTEM

The DSNTYPE value from the SMS data class is used. If no SMS data class is defined, or if it does not specify the DSNTYPE value, the system DSNTYPE value is used. This is the default value.

BASIC

Allocates physical sequential data sets as physical sequential basic format data sets.

LARGE

Allocates physical sequential data sets as physical sequential large format data sets.

DSWAITTIME

Indicates the number of minutes the FTP server waits for an MVS data set to become available when a local data set is held by another job or process. The value 0 indicates that the FTP server does not wait to obtain a data set when the data set is being held by another job or process.

EATTR

Indicates whether newly allocated data sets can have extended attributes and whether new data sets can reside in the EAS of an EAV.

SYSTEM

The data set uses the SMS data class EATTR value. If no SMS data class is defined, or if the data class contains no EATTR specification, the data set is allocated with the system default.

NO

The data set cannot reside in the EAS, and its VTOC entry cannot contain extended attributes.

OPT

The data set can reside in the EAS, and its VTOC entry can have extended attributes if the volume supports them.

ENCODING

Indicates the encoding type that is used for conversions between code pages for data transfers.

FIFOIOTIME

Indicates the length of time the that FTP server waits for a read from a z/OS UNIX named pipe or a write to a z/OS UNIX named pipe to complete.

FIFOOPENTIME

Indicates the length of time that the FTP server waits for an open of a z/OS UNIX named pipe to complete.

FILEtype

Indicates the data set file type.

FTpkeepalive

Indicates the control connection keepalive timer value in seconds.

INactivetime

Indicates the inactivity timer to a specified number of seconds.

ISPFSTATS

Indicates that FTP will create or update ISPF Member statistics when PUT, MPUT, or APPEND subcommands are issued.

JESENTRYLimit

Indicates the number of entries that can be displayed concurrently using a LIST or NLST command.

JESGETBYDSN

Indicates whether the server should retrieve the file from the MVS system and submit it as a batch job when FILETYPE is JES and JESINTERFACELEVEL is 2, or whether the server should retrieve the JES spool file by the data set name.

JESJOBName

Indicates that any command (Get, LIST, DIR, or MGET) should be limited to those jobs, started tasks, APPC/MVS, or TSO output that match the specified value.

JESLRECL

Indicates the logical record length (LRECL) for the Job Entry System (JES) internal reader at the foreign host.

JESOWNER

Indicates that any command (Get, LIST, DIR, or MGET) should be limited to those jobs, started tasks, APPC/MVS, or TSO output which are owned by the user ID specified.

JESRECFM

Indicates the record format for the JES internal reader at the foreign host.

JESSTATUS

Indicates what type of information should be returned on LIST and NLST commands.

LISTLEVEL

Indicates which format the FTP server will use when it replies to the LIST command.

LISTSUBDIR

Indicates that wildcard searches should apply to the current working directory and should also span its subdirectories.

LRECL

Indicates the logical record length (LRECL) of a newly allocated data set.

MBDATACONN

Indicates the code pages for the file system and for the network transfer that are used when the server does data conversion during a data transfer.

MBREQUIRELASTEOL

Indicates whether the FTP server reports an error when a multibyte file or data set is received from the server with no EOL sequence in the last record received.

MBSSENDEOL

Indicates which end-of-line sequence to use when the ENCODING value is SBCS, the data is ASCII, and data is being sent to the server.

MGMTCLASS

Indicates the SMS management class as defined for the target host by your organization.

MIGRATEVOL

Indicates the volume ID for migrated data sets if they do not use IBM storage management systems.

PDSTYPE

Indicates whether the FTP server creates local MVS directories as partitioned data sets or as partitioned data sets extended.

PRIMARY

Indicates the number of tracks, blocks, or cylinders for the primary allocation.

QUOTEVERRIDE

Indicates that a single quotation mark at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory.

RDW

Indicates that variable record descriptor words (RDWs) are treated as if they are part of the record and are not discarded during FTP transmission of variable format data sets in stream mode.

READTAPEFormat

Displays information about an input data set on tape.

RECFm

Displays the data set record format.

RETpd

Indicates the number of days that a newly allocated data set should be retained.

SBDataconn

Indicates the conversions between file system and network code pages to be used for data transfers.

SBSENDEOL

Indicates which end-of-line sequence to use when ENCODING is SBCS, the data is ASCII, and data is being sent to the client.

SBSUB

Indicates that substitution is allowed for data bytes that cannot be translated in a single-byte-character translation.

SBSUBCHAR

Indicates the value that is used for substitution when SBSUB is also specified.

SECondary

Indicates the number of tracks, blocks, or cylinders for the secondary allocation.

SECUREIMPLICITos

Indicates that when the client connects using the TLSPORT implicit connection, the client waits for the 220 good morning reply before initiating the security handshake and negotiation.

SPRead

Indicates that the output is in spreadsheet format when the file type is SQL.

SQLCol

Indicates the SQL output file column headings.

STOrclass

Indicates the SMS storage class as defined by your organization for the target host.

TLSRFCLEVEL

Indicates the level of RFC 4217, *On Securing FTP with TLS*, that is supported by the server.

TRacks

Indicates that primary and secondary space allocations are in tracks.

TRAILingblanks

Indicates whether the FTP server preserves the trailing blanks in a fixed-format data set when the data is sent to a foreign host.

TRUNCate

Indicates that truncation is permitted.

UCOUNT

Indicates the number of devices to allocate concurrently to support the allocation request.

UCSHOSTCS

Indicates the EBCDIC code set to be used when converting to and from Unicode.

UCSSUB

Indicates that in Unicode-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any Unicode character that cannot be successfully converted.

UCSTRUNC

Indicates that in Unicode-to-EBCDIC conversion, EBCDIC data truncation is allowed.

UMask

Indicates the file mode creation mask.

UNICODEFILESYSTEMBOM

Indicates whether the FTP server will store incoming Unicode files with a byte order mark.

Unit

Indicates the unit type for allocation of new data sets.

UNIXFILETYPE

Indicates whether the FTP server treats files in its z/OS UNIX file system as regular files or as named pipes.

VCOUNT

Indicates the number of tape data set volumes that an allocated data set can span.

VOLUME

Indicates the volume serial number for allocation of new data sets.

WRAPRecord

Indicates that data is wrapped to the next record if no new-line character is encountered before the logical record length of the receiving file is reached.

WRTAPEFastio

Indicates that ASCII stream data that is being written to tape can be written using BSAM I/O.

XLate

Indicates the translate table to be used for the data connection.

Examples

The following is an example of the STatus subcommand using a single parameter:

```
status (asatrans
>>> XSTA (asatrans
211-ASA control characters in ASA files opened for text processing
211-will be transferred as ASA control characters.
211 *** end of status ***
```

The following is an example of retrieving the status information from an FTP server:

```

status
>>>STAT
211-Server FTP talking to host 9.117.222.59, port 23467
211-User: USER33 Working directory: /u/user33
211-The control connection has transferred 395 bytes
211-There is no current data connection.
211-The next data connection will be actively opened
211-to host 9.117.222.59, port 23467,
211-using Mode Stream, Structure File, type EBCDIC, byte-size 8
211-Automatic recall of migrated data sets.
211-Automatic mount of direct access volumes.
211-Auto tape mount is allowed.
211-Inactivity timer is disabled
211-Server site variable DSWAITTIME is set to 10
211-Server site variable DATAKEEPAIVE is set to 120
211-VCOUNT is 59
211-ASA control characters in ASA files opened for text processing
211-will be transferred as ASA control characters.
211-Trailing blanks are removed from a fixed format
211-data set when it is retrieved.
211-Data set mode. (Do not treat each qualifier as a directory.)
211-ISPFSTATS is set to FALSE
211-Primary allocation 2 tracks. Secondary allocation 1 track.
211-Partitioned data sets will be created with 27 directory blocks.
211-FileType SEQ (Sequential - default).
211-Number of access method buffers is 5
211-RDWs from variable format data sets are discarded.
211-Records on input tape are unspecified format
211-SITE DB2 subsystem name is D7A
211-Data not wrapped into next record.
211-Tape write is not allowed to use BSAM I/O
211-Truncated records will not be treated as an error
211-JESLRECL is 80
211-JESRECFM is Fixed
211-JESINTERFACELEVEL is 1
211-ENcoding is set to SBCS
211-DBSUB is set to TRUE
211-SBSUB is set to FALSE
211-SBSUBCHAR is set to SPACE
211-SMS is active.
211-New data sets will be catalogued if a store operation ends abnormally
211-Single quotes will override the current working directory.
211-UMASK value is 027
211-Process id is 50331660
211-Checkpoint interval is 0
211-Authentication type: None
211-Record format FB, Lrecl: 80, Blocksize: 3120
211-Server site variable EATTR is set to OPT
211-Server site variable DSNTYPE is set to LARGE
211-Server site variable LISTSUBDIR is set to TRUE
211 *** end of status *** Command:

```

Usage

The retrieved status information can be a directory, a file, or general status information, such as a summary of activity. If *name* is omitted, general status information is retrieved.

For further information about setting values for server initialization in the FTP.DATA data set, see the [z/OS Communications Server: IP Configuration Reference](#).

STREAm subcommand—Set the stream data transfer mode

Purpose

Use the STREAm subcommand to set the data transfer mode to stream mode. This is equivalent to specifying the MMode S subcommand. See [“MMode subcommand—Set the data transfer mode”](#) on page 220 for more information.

Format

►► STREAm ◄◄

Parameters

There are no parameters for this subcommand.

STRucture subcommand—Set the file structure

Purpose

Use the STRucture subcommand to set the file structure.

Format

►► STRucture — File — ◄◄
 └── Record ─┘

Parameters

File

Sets the file structure to File. When the STRucture value is File, the file is sent as a continuous sequence of data bytes.

Record

Sets the file structure to Record. When the STRucture value is Record, the file is sent as a series of records.

Tip: Use STRucture RECORD in conjunction with a CHKConfidence value of TRUE in FTP.DATA to improve detection of incomplete file transfers.

SUnique subcommand—Changes the storage method

Purpose

The SUnique subcommand changes the method of storing files on the foreign host.

Format

►► SUnique — On — Off — NAME — NONAME — ◄◄

Parameters

On

Turns on store unique.

Off

Turns off store unique.

NAME

When specified with ON or OFF, instructs the client to include a name when sending a store-unique command to the server.

NONAME

When specified with ON or OFF, instructs the client to omit a name when sending the store-unique command to the server.

Usage

- By default, the SUnique setting is OFF NAME and FTP uses a store command (STOR) with the PUt and MPut subcommands. If the foreign host already has a data set or file with the name specified by the *foreign_file* value, the foreign host overwrites the existing data set or file.
- If SUnique is set to ON, FTP uses a store-unique command (STOU) with the PUt and MPut subcommands and prevents you from overwriting or erasing the existing data set or file on the foreign host. If the default setting of NAME is in effect, a name string is sent to the server with the store-unique command. The created foreign data set or file is stored with a unique name. FTP sends the unique name of the created foreign data set or file to the local host, where the data set or file name is displayed on your terminal.
- SUnique with no parameters toggles the ON/OFF setting. If ON or OFF is specified, SUnique is set to that value, regardless of its current setting. The NAME/NONAME setting can be changed as SUnique is turned ON or OFF. It is in effect when SUnique is ON and does not change for the session until another NAME or NONAME setting is specified.

SYstem subcommand—Display the operating system name

Purpose

Use the SYstem subcommand to display the name of the remote host operating system. The remote host must have also implemented the SYST subcommand.

Format

►► SYstem ◄◄

Parameters

There are no parameters for this subcommand.

Usage

Use this subcommand to determine the operating system at the server. The reply from the server has as its first word one of the system names from the Protocol Numbers and Assignment Services list for Operating System Names. The names are maintained by the Internet Assigned Numbers Authority (<http://www.iana.org>). For the z/OS server, the reply is determined by the current working directory at the server.

- If the current working directory is a z/OS UNIX file system directory, the reply is:

```
215 UNIX is the operating system of this server. FTP Server is running on z/OS.
```

- Otherwise, the reply is:

```
215 MVS is the operating system of this server. FTP Server is running on z/OS.
```

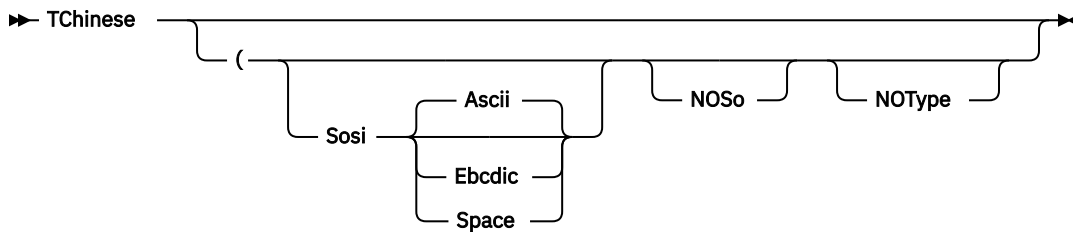
TChinese subcommand—Change the data transfer type to TCHINESE

Purpose

Use the TChinese subcommand to change the data transfer type to Traditional Chinese (5550).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the `Site/LOCSite SBDataconn` or `Site XLate` subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi

Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — `Ascii`, `Ebcdic`, or `Space`. If no parameter is specified, ASCII is used as the default.

If the `S` parameter is not specified, shift-out or shift-in characters are not used in the transferred data.

Ascii

When combined with the `Sosi` parameter, causes shift-out and shift-in characters `X'1E'` and `X'1F'` to be used to delimit DBCS strings in ASCII data.

Ebcdic

When combined with the `Sosi` parameter, causes shift-out and shift-in characters `X'0E'` and `X'0F'` to be used to delimit DBCS strings in ASCII data.

Space

When combined with the `Sosi` parameter, causes shift-out and shift-in characters `X'20'` and `X'20'` (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo

Specifies that the data transferred is pure DBCS (data with no SBCS characters) and is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType

Suppresses the sending of the corresponding `Type` command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The `TChinese` client subcommand is equivalent to the `TYPE B 7` server command.

Context

See [“FTP with traditional DBCS support” on page 78](#) and [“Support for MBCS languages” on page 81](#) for more information.

TSO subcommand—Use TSO commands

Purpose

Use the `TSO` subcommand to pass a Time Sharing Option (TSO) command to a local host TSO environment.

Format

► TSO — *command_line* ◄

Parameters

command_line

Specifies a TSO command. Do not use synonyms.

Usage

The TSO subcommand is not available from batch.

Restrictions

You cannot use TSO commands that run with POSIX(ON) because of a Language Environment restriction with nested enclaves. See [Using nested enclaves in z/OS Language Environment Programming Guide](#) for more information. The following TSO commands run with POSIX(ON):

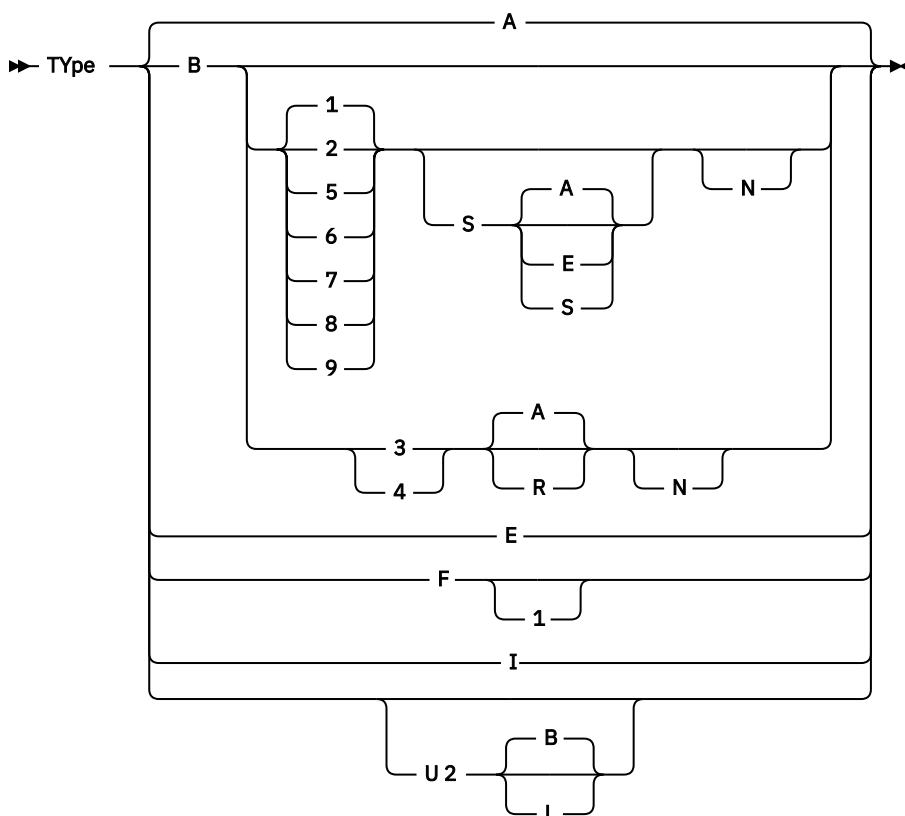
- FTP
- PING
- NETSTAT
- TRACERTE

TType subcommand—Set the data transfer type

Purpose

Use the TType subcommand to set the data transfer type for the client and server simultaneously with one command. FTP supports the ASCII, EBCDIC, image (binary), UCS-2, and two DBCS data transfer types.

Format



Parameters

A

Sets the transfer type as ASCII. Specifying the ASCII transfer type has the same effect as using the AScii subcommand. When the data transfer type is ASCII, FTP translates outgoing files to ASCII before sending them, and translates incoming files from ASCII to the file system code page before storing them. ASCII is the default transfer type.

B

Sets the transfer type as DBCS. Specifying the B transfer type with the appropriate options has the same effect as using the BIG5, EUckanji, HAngeul, JIS78kj, JIS83kj, Ksc5601, SJiskanji, SChinese, or TChinese subcommands. If B is specified alone, the second type parameter defaults to 1 and current transfer type is changed to Shift JIS kanji.

When you transfer double-byte data, the currently active SBCS translation table is used for SBCS characters in the data set. If necessary, use the SItE/LOCSItE SBDataconn or SItE XLate FTP subcommand to select an alternate SBCS translation table that is appropriate for your data before transferring your double-byte data.

B 1

Changes current transfer type to Shift JIS kanji.

B 2

Changes current transfer type to Extended UNIX Code kanji.

B 3

Changes current transfer type to JIS 1983 kanji.

B 4

Changes current transfer type to JIS 1978 kanji.

B 5

Changes current transfer type to Hangeul.

- B 6**
Changes current transfer type to Korean Standard Code KSC-5601, 1989 version.
- B 7**
Changes current transfer type to Traditional Chinese (5550).
- B 8**
Changes current transfer type to Big-5.
- B 9**
Changes current transfer type to Simplified Chinese.
- S**
Transferred data contains shift-out and shift-in delimiters.
If S is specified alone, the second parameter defaults to A. Shift-out and shift-in characters X'1E' and X'1F' are used.
The S parameter can be used to control the use of shift-out (SO) and shift-in (SI) characters during DBCS data transfer for Big5, SChinese, Shift-JIS kanji, EUC kanji, Hangeul, KSC-5601, and TChinese.
If Sosi is not specified at all, shift-out or shift-in characters are not used in the transferred data.
- S A**
Use shift-out and shift-in characters X'1E' and X'1F' to delimit DBCS strings in the transferred data.
- S E**
Use shift-out and shift-in characters X'0E' and X'0F' to delimit DBCS strings in the transferred data.
- S S**
Use ASCII spaces (X'20') as shift-out and shift-in characters to delimit DBCS strings in the transferred data.
- A**
Use ASCII shift-in escape sequence ESC (B. This is the default. (Used for DBCS data types JIS 1983 kanji and JIS 1978 kanji only.)
- R**
Use JISROMAN shift-in escape sequence ESC (J. (Used for DBCS data types JIS 1983 kanji and JIS 1978 kanji only.)
- N**
Indicates that the transfer is to be pure DBCS data (data with no SBCS characters) and that the data is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.
When data is transferred from the EBCDIC host, the entire data set is assumed to be EBCDIC DBCS with no SO/SI characters in the data. The data is then converted to the required ASCII type and if any SO/SI option has been specified for the transferred data then the corresponding SO/SI characters are used to delimit the ASCII DBCS strings.
When transferring data to the EBCDIC host, no SO/SI characters are inserted, and if any SO/SI option is specified for the transferred data, the corresponding SO/SI characters are removed from the ASCII data and not replaced at the host. The length of data might change during transfer to and from the EBCDIC host when pure DBCS is specified with any SO/SI option. When pure DBCS is specified by itself, the length of data does not change. If N is not specified, the shift-out or/shift-in characters X'0E' and X'0F' are used at the host.
- E**
Sets the transfer type as EBCDIC. Specifying the EBCDIC transfer type has the same effect as using the Ebcdic subcommand. The EBCDIC transfer type is intended for efficient transfer between hosts that use EBCDIC for their internal character representation.
- F**
Sets the transfer type as EBCDIC IBM kanji. Specifying the IBM kanji transfer type has the same effect as using the Ibmkanji subcommand.

F 1

Change current transfer type to IBM (EBCDIC) kanji.

I

Sets the transfer type as image (binary). Specifying the image transfer type has the same effect as using the BINary subcommand. With the image transfer type, data is sent as contiguous bits, packed into 8-bit bytes. The image transfer type is used for the efficient storage and retrieval of data sets or files, and for the transfer of binary data.

U 2

Sets the transfer type to Unicode UCS-2. TYPe U 2 has optional parameters:

B

Specifies big-endian byte order for Unicode encoding. This is the default.

L

Specifies little-endian byte order for Unicode encoding.

Examples

- Transfer text data to another host:

```
User:      ascii
System:    >>>TYPE A
           200 Representation type is ASCII.
Command:
```

- Transfer binary data to another host:

```
User:      type i
System:    >>>TYPE I
           200 Representation type is IMAGE.
Command:
```

- Transfer text data from an EBCDIC host to an EBCDIC host:

```
User:      type e
System:    >>>TYPE E
           200 Representation type is EbcDic NonPrint
Command:
```

- Transfer binary data from an EBCDIC host to an EBCDIC host:

```
User:      type i
System:    >>>TYPE I
           200 Representation type is Image.
Command:
```

- Set the transfer type to JIS 1983 kanji using the JISROMAN shift-in escape sequence ESC (J:

```
TYPE B 3 R
```

- Set the transfer type to Shift-JIS kanji using the EBCDIC SO/SI characters X'0E'/X'0F' in the transferred date:

```
TYPE B 1 S E
```

Usage

- If no Sosi option is specified by the TYPe command for BIG5, SChinese, EUckanji, HAngeul, Ksc5601, SJiskanji, or TChinese, standard DBCS control is used for the data transfer. This means that no SO/SI characters are placed in the ASCII data when transferring from the (EBCDIC) host to ASCII and the value of each ASCII character is used to determine if it is a single-byte character or part of a double-byte character when transferring to the host. For JIS 1983 kanji and JIS 1978 kanji, three-character

escape sequences are always used to delimit DBCS strings in mixed SBCS/DBCS ASCII data. These escape sequences cannot be altered by using the S, S A, S E, or S S parameters.

- If no Sosi option is specified, the length of data might change as it is transferred to or from the EBCDIC host since EBCDIC DBCS types on the host contain SO/SI characters in mixed SBCS/DBCS data to determine which characters are part of a DBCS string. Any of the above SO/SI options (S, S A, S E or S S) can be used for mixed SBCS/DBCS data so that the length of data does not change when transferred to or from the EBCDIC host. Use of three-character escape sequences for JIS 1983 kanji and JIS 1978 kanji means that the length of data for these types always changes when transferring mixed SBCS/DBCS data to or from the EBCDIC host.
- Use ASCII spaces as SO/SI characters in the transferred data only for transfer from the EBCDIC host. Data can be transferred to the host when using this option but care must be taken as each ASCII space is interpreted as a shift-out or shift-in character and is replaced with the corresponding SO/SI character on the host.

Context

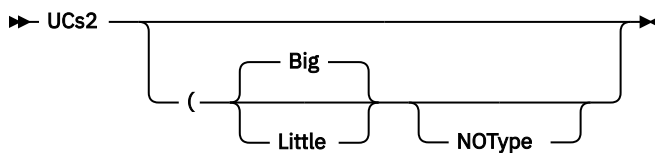
For more information about transfer methods, see [Table 13 on page 46](#).

UCs2 subcommand—Change data transfer type to Unicode UCS-2

Purpose

Use the UCs2 subcommand to change the data transfer type to Unicode UCS-2.

Format



Parameters

Big

Specifies big-endian byte order for the Unicode encoding. This is the default.

Little

Specifies little-endian byte order for the Unicode encoding.

NOType

Suppresses the sending of the corresponding TType command to the server. Use this parameter when you want translation to be done by the FTP client only.

Usage

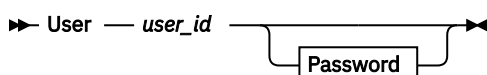
The UCs2 client subcommand is equivalent to the TType U 2 subcommand.

User subcommand—Identify yourself to a host or change your TSO user ID password

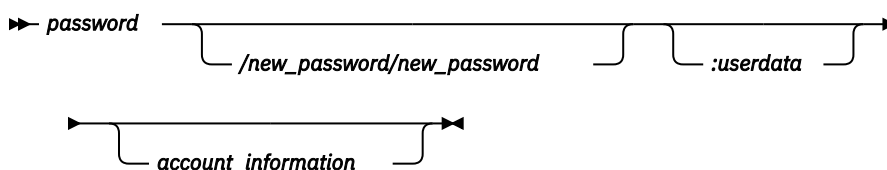
Purpose

Use the User subcommand to identify yourself to the remote host after opening a connection. If the remote host is a z/OS FTP server, you can change your TSO user ID, password, or password phrase.

Format



Where Password is



Parameters

user_id

Specifies your login name on the host.

password

Specifies your current password or password phase on the host. If you do not supply *password* when invoking the User subcommand, you are prompted to enter a password if the host requires a password to log in.

new_password/new_password

An optional parameter that specifies your new password or password phase on the host. You must enter the password twice.

Requirement: SAF-compliant security products such as RACF require that *password* and *new_password* both be passwords, or both be password phrases.

:userdata

The optional user data must be separated from the password information by a colon (:) and can be any combination of up to 200 nonblank characters and numbers—except the colon. Care should be taken when using the back slash character (\) in combination with other characters, which might be interpreted as an escape sequence by the C compiler.

account_information

An optional parameter that will be supplied to the remote FTP server if the server requests account information after receiving the password.

Results:

- Not all FTP servers support the *userdata* parameter. The z/OS FTP server interprets *userdata* as a character string and passes it to the server FTCHKPWD user exit routine.
- If you enter your password or password phrase incorrectly, the client does not prompt you to enter the password again. You must reissue the User subcommand to enter the correct password.
- If you do not specify *password/new_password/new_password* on the User subcommand, you can specify it when you are prompted for the password after entering the User subcommand. You can issue the User subcommand to change your TSO *user_id* password at any time during the FTP session.

Tips:

- To avoid having your password print when coding your user ID and password as part of a CLIST or batch job, enter your user ID and password on separate lines.
- You can use the NETRC data set to automatically provide the user ID, password and account information to log in to an FTP server.

Rules:

- Do not place any spaces between the passwords and the slashes (/), and the user data.

- Enter a password phrase that contains blanks by enclosing the entire password phrase in quotation marks. You can use single or double quotation marks. If the password phrase itself contains a quotation mark, use the other style of quotation marks to enclose the password phrase.

Example: Enter the phrase *What's up, Doc?* as *"What's up, Doc?"*, but not as *'What's up, Doc?'*.

This rule applies also to the account information and user IDs.

- Do not use quotation marks to enclose a password phrase that is comprised only of any of the following characters:
 - Uppercase or lowercase letters
 - Numerals from 0 to 9
 - The following special characters:
 - @
 - #
 - \$
 - -
 - {
 - .
 - (
 -)
 - *
 - %
 - +

This rule applies also to user IDs and to the account information.

Example: Enter the password phrase *JoeIBMer@ibm.com* as *JoeIBMer@ibm.com*, but not as *'JoeIBMer@ibm.com'*, nor as *"JoeIBMer@ibm.com"*.

Note: When you use FTP through a proxy that requires two passwords with one for the firewall and the other for the remote system, specify both passwords by enclosing them in quotation marks.

Example: Enter the user IDs NAME1 and NAME2 as *'NAME1 NAME2'*, or as *"NAME1 NAME2"*.

Restrictions:

- A password phrase, user ID, or the account information that you enter at the z/OS FTP client must not contain both single quotation mark and double quotation mark characters. You can use either style of quotation marks in the user ID, password phrase, or account information, but not both.

Example: The password phrase *What's up, Doc?* is valid because it contains only single quotation marks. You enter it at the z/OS FTP client as *"What's up, Doc?"*. The phrase *"What's up, Doc?"* with the double quotation marks as part of the phrase cannot be entered at the z/OS FTP client because it contains both styles of quotation marks.

If you enter the *password/new_password/new_password* argument, the sequence *password/new_password* cannot contain both single quotation mark and double quotation mark characters. You can use either style of quotation marks, but not both.

Example: The password *What's up, Doc?* and new password *Not much; what's up with you?* are valid because the two password phrases contain only single quotation marks. You enter it at the z/OS FTP client as *"What's up, Doc?/Not much; what's up with you?/Not much; what's up with you"*. The password phrases *"What's up, Doc? "* and *He said, "not much; you?"* cannot be entered as a *password/new_password/new_password* sequence at the z/OS FTP client because the password phrases use both styles of quotation marks.

- When entering this subcommand in a USS environment, you can enter only up to 510 characters including the subcommand name. When entering the optional password argument as *password/*

newpass/newpass:userdata account_information, such that *password* and *newpass* are password phrases, you must take this into account.

Related Topic: See “NETRC data set” on page 28 for information about using NETRC.

Verbose subcommand - Toggle verbose mode

Purpose

Use the Verbose subcommand to toggle verbose mode. When in verbose mode the client displays message IDs. This subcommand is effective only when the client is running in a z/OS UNIX environment.

Format

►► Verbose ◄◄

Parameters

There are no parameters for this subcommand.

Examples

```
verbose
Message IDs are not displayed when running in z/OS UNIX
Command:
lpw
Local directory name set to hierarchical file /u/user33
Command:
verbose
EZA2859I Message IDs are displayed when running in z/OS UNIX
EZA1460I Command:
lpw
EZA2578I Local directory name set to hierarchical file /u/user33
EZA1460I Command:
```

Usage

Verbose is normally toggled off when the client starts. If you want it toggled on when the client starts, use the `-v` parameter on the FTP command. If the FTP client is running in a TSO environment, the display of message IDs is controlled by the profile options MSGID and NOMSGID.

Chapter 6. Sending electronic mail using the Communications Server SMTP application

The Communication Server SMTP (CSSMTP) application is a mail forwarding SMTP client. CSSMTP processes data sets that are in the JES spool file that contain mail messages and then forwards the mail messages to a target server. This topic describes how to prepare mail messages, add them to the JES spool data set, and forward them to the target server.

This topic describes the following information:

- Creating mail messages on the JES spool data set
 - [“Using the SMTPNOTE command” on page 297](#)
 - [“Using the TSO TRANSMIT command to send a mail file” on page 300](#)
 - [“Using the IEBGENER utility to copy a mail file to a JES sysout file” on page 301](#)
- [“SMTP commands” on page 302](#)
- [“SMTP commands and reply codes across a TCP/IP connection” on page 308](#)
- [“CSSMTP exit for unwanted mail” on page 308](#)
- [“Example of receiving mail” on page 308](#)
- [“Example of an undelivered mail notification” on page 309](#)
- [“Example of generated error reports” on page 309](#)

Creating mail messages on the JES spool data set

This topic describes how to prepare mail messages and add them to the JES spool data set.

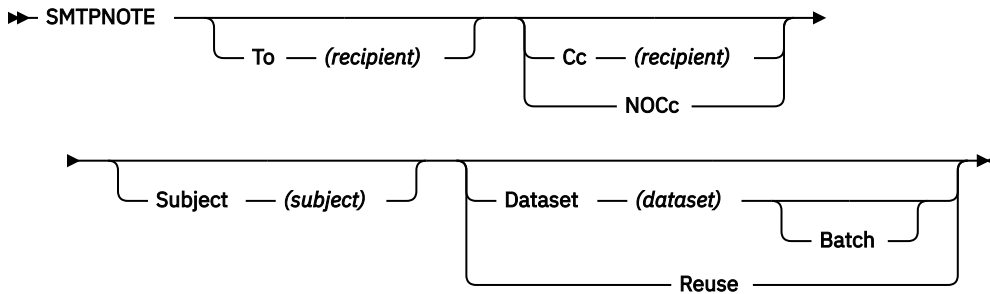
Using the SMTPNOTE command

Purpose

Use the SMTPNOTE command to prepare mail for one or more recipients using the Time Sharing Option (TSO) EDIT command or to send mail that is created with another system editor.

Send electronic mail to one or more recipients on TCP networks using the SMTPNOTE command. Use the SMTPNOTE command to transmit mail messages to the JES spool data set to be processed by the CSSMTP application. If the CSSMTP application does not have an ExtWrtName statement, you must customize the SMTPJOB variable in the SMTPNOTE CLIST to match the CSSMTP application job name; otherwise, ensure that the SMTPJOB variable in the SMTPNOTE CLIST matches the value that is specified on the ExtWrtName configuration statement. The DOMAIN variable should also be set in the SMTPNOTE CLIST when you are using CSSMTP. For more information about customizing the SMTPNOTE CLIST, see the steps for [customizing the SMTPNOTE CLIST \(optional\)](#) in [z/OS Communications Server: IP Configuration Guide](#).

Format



Parameters

To (*recipient*)

Specifies a single recipient for the mail message. If you do not specify the To parameter, you are prompted to enter a list of recipients. Enter the name of each recipient on a separate line and end the recipient list with a blank line. You must specify at least one mail recipient.

Cc (*recipient*)

Specifies a single copy recipient for the mail message. If you do not specify the Cc parameter and you do not specify the NOCc or Batch parameters, you are prompted to enter a list of copy recipients.

If there are no copy recipients, press **Enter**; otherwise, enter the name of each recipient on a separate line and end the list with a blank line.

NOCc

Specifies that no prompting for the Cc parameter takes place.

(*recipient*)

Specifies the path address of the mail recipient. The format of the recipient parameter is equivalent to the path syntax, as described in RFC 2821, without the less than (<) and greater than (>) delimiters. For information about accessing RFCs, see [Appendix D, “Related protocol specifications,”](#) on page 381.

The *recipient* parameter has one of the following formats:

user_id@host_name

A user on a host in your domain, possibly on your local node.

user_id@host_name.domain

A user on a host in a specified domain.

Restrictions:

- The CSSMTP application does not rewrite the specified path name. The batch job must specify the proper email address formats, such as *userid@host.domain*. For example, in the command string *user_id%nje_host_name@gateway_name.domain*, the term *user_id%nje_host_name* is treated as a user ID.
- The CSSMTP application does not support source routing, such as *@host1,@host2:userid@host3* or *NJEuserid%NJEhost* format. For example, in the address string *@host1,@host2,...,@hostn:user_id@host_name*, the *@host1,@host2,...,@hostn* portion of the address is ignored and the *user_id@host_name* portion of the address is used as the recipient value.
- The CSSMTP application does not generate source routing addressing formats.

Subject (*subject*)

Specifies the subject of the mail message. If the subject is specified as a keyword, the message cannot contain embedded blanks; otherwise, the subject can be any character string. The maximum length is 233 characters.

Batch

Specifies that no prompting takes place. You must also specify the To and Dataset parameters.

Dataset (*data_set_name*)

Indicates that the text of the mail is contained in a sequential data set. The data set can have any record format, can be blocked or unblocked, and can have records that are 1 – 243 characters in length. The *data_set_name* variable is the name of the sequential data set that contains the mail message text. The data set name must be a valid data set name, and is fully qualified if it is within single quotation marks (').

Reuse

Causes SMTP to reuse the contents of a mail message that was previously canceled. If a mail message was not canceled, the Reuse parameter is ignored.

Guidelines:

- When the To, Cc, Subject, or Dataset parameters are specified more than once, the SMTPNOTE application uses the last value that was specified. No error messages are generated if there are duplicate parameters on the command line.
- SMTPNOTE does not require quotation marks around blanks, or single quotation marks, semicolons, or commas around data set names.

Restriction: Do not use extended attribute data sets (PDSE) with the SMTPNOTE command.

Examples

After you enter the SMTPNOTE command, you are prompted for the mail recipients (To:), the copy recipients (Cc:), and the subject of the mail message (Subject:), if they were not specified with the SMTPNOTE command. If you enter a list of recipients, enter the name of each recipient on a separate line and enter a blank line to indicate the end of the list.

After you answer the prompts, SMTPNOTE invokes the TSO EDIT command to enable you to prepare your mail message. Typically, the TSO editor starts in INPUT mode. Enter the text of your mail message line by line. When your mail message is complete, enter a null line (that is, do not type anything when you are prompted) and press **Enter**. The editor switches to EDIT mode.

In EDIT mode, you can use all of the functions of the editor. You can also return to INPUT mode, send the mail message, or cancel the mail message. For a complete description of the EDIT command, see [z/OS TSO/E Command Reference](#).

If you invoke the SMTPNOTE command with the Reuse or Dataset parameter, your session is immediately placed in EDIT mode. The contents of the previously canceled mail message or the data set that you specify are already part of the mail message. You can add to or change the existing data.

To send the mail message, enter END SAVE in EDIT mode, and then enter SEND. To cancel the mail message, enter END SAVE or END NOSAVE in EDIT mode, and then enter CANCEL. If you cancel a mail message, you can recover the data that you entered by invoking the SMTPNOTE command with the Reuse parameter. The recipients and subject of the mail message are not saved, and must be reentered. The following is an example of the results of preparing and sending mail.

```

READY
smtpnote
TO:
irvine@mvs2.accounting
bekker@mvs2.accounting
mcgregor@mvs1.accounting

CC:

SUBJECT:
Travel Expenses

ENTER "END SAVE" TO SAVE THE NOTE.
FOR A COMPLETE LIST OF EDIT SUBCOMMANDS ENTER "HELP".
INPUT
Could we please postpone the expense review because I
will be out of town the week of the 19th.

How about the 23rd? Thank you, John.
```

```

EDIT
end save
ENTER "SEND" TO SEND THE NOTE.
ENTER "CANCEL" TO TERMINATE WITHOUT SENDING THE NOTE.
send
READY

```

Using the TSO TRANSMIT command to send a mail file

Purpose

Use a TSO TRANSMIT (XMIT) command to send to the spool file a previously constructed mail file that contains SMTP commands to be processed by CSSMTP for one or more mail messages. See [“SMTP commands”](#) on page 302 for more information about each SMTP command.

Examples

1. Construct a mail file named MYCSSMTP.NOTE:

```

HELO YOURMVS
MAIL FROM:<CAROL@YOUR.MVSDOMAIN.COM>
RCPT TO:<msgs@rsch.our.edu>
RCPT TO:<alice@ai.our.edu>
DATA
Date: Thur, 26 Mar 92 21:48:57 EST
From: Carol <CAROL@YOUR.MVSDOMAIN.COM>
To: <msgs@rsch.our.edu>
Cc: <alice@ai.our.edu>
Subject: update

Mike: Cindy stubbed her toe. Bobby went to
      baseball camp. Marsha made the cheerleading team.
      Jan got glasses. Peter has an identity crisis.
      Greg made dates with 3 girls and couldn't
      remember their names.

.
QUIT

```

2. Use the XMIT command to put the file *userid.MYCSSMTP.NOTE* into the JES spool data set:

```
XMIT jesnode.cssmtp1 DA(userid.mycssmtp.note)
```

jesnode

The JES node name or host name.

cssmtp1

The external writer name if the ExtWrtName statement is specified in the CSSMTP application configuration file, or the CSSMTP application address space name if the ExtWrtName statement is not specified in the CSSMTP application configuration file.

userid.mycssmtp.note

The SMTP commands data set.

Note: If you send a member of a partitioned data set (PDS) using the TSO TRANSMIT command, the member must be converted to a sequential file. Use the SEQ option on the TSO TRANSMIT (XMIT) command as shown in the following example:

```
XMIT jesnode.cssmtp1 DA(pds(member)) SEQ
```

In this example, *pds* is the name of the partitioned data set and *member* is the name of the PDS member that is to be sent to the CSSMTP1 external writer name.

Using the IEBGENER utility to copy a mail file to a JES sysout file

Purpose

Use the IEBGENER utility to copy a previously constructed mail file to the JES spool data set that contains SMTP commands for one or more mail messages for the CSSMTP application to process. See [“SMTP commands”](#) on page 302 for more information about each SMTP command.

Examples

- Construct a mail file named *userid.MYCSSMTP.NOTE*:

```
HELO YOURMVS
MAIL FROM:<CAROL@YOUR.example.COM>
RCPT TO:<msgs@rsch.example.edu>
RCPT TO:<alice@ai.example.edu>
DATA
Date: Thur, 26 Mar 92 21:48:57 EST
From: Carol< CAROL@YOUR.example.COM>
To: <msgs@rsch.example.edu>
Cc: <alice@ai.example.edu>
Subject: update

    Mike: Cindy stubbed her toe.  Bobby went to
          baseball camp.  Marsha made the cheerleading team.
          Jan got glasses.  Peter has an identity crisis.
          Greg made dates with 3 girls and couldn't
          remember their names.

.
QUIT
```

- Create JCL using the IEBGENER utility:

```
//jobname JOB (accounting.information),"programmer.name",CLASS=A,MSGCLASS=A,
//          NOTIFY=userid
//COPY     EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DISP=SHR,DSN=userid.MYCSSMTP.NOTE
//SYSUT2   DD SYSOUT=(A,CSSMTP1),SPIN=ALLOC
```

- Create JCL with inline input for SYSUT1 and SYSUT2 using the IEBGENER utility:

```
//jobname JOB (accounting.information),"programmer.name",CLASS=A,MSGCLASS=A,
//          NOTIFY=userid
//COPY     EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD *
HELO YOURMVS
MAIL FROM:<CAROL@YOUR.example.COM>
RCPT TO:<msgs@rsch.example.edu>
RCPT TO:<alice@ai.example.edu>
DATA
Date: Thur, 26 Mar 92 21:48:57 EST
From: Carol <CAROL@YOUR.example.COM>
To: <msgs@rsch.example.edu>
Cc: <alice@ai.example.edu>
Subject: update

Mike: Cindy stubbed her toe.  Bobby went to
baseball camp.  Marsha made the cheerleading team.
Jan got glasses.  Peter has an identity crisis.
Greg made dates with 3 girls and couldn't
remember their names.

.
QUIT
//SYSUT2   DD SYSOUT=(A,CSSMTP1),SPIN=UNALLOC
```

Rules:

- If necessary, code the DCB attributes on the SYSUT2 DD statement. The following example shows how to code the DCB attributes:

```
//SYSUT2 DD SYSOUT=(A,CSSMTP1),DCB=(LRECL=133,BLKSIZE=27930),FREE=CLOSE
```

The logical record limit for files is 1024 bytes. The block size limit for files that are placed on the spool file is 32760.

- Do not place spanning record files on the spool file for the CSSMTP application to use, because the CSSMTP application does not support these files.
- Use SPIN=UNALLOC or CLOSE=FREE when you create multiple sysout files in one job. This creates a separate sysout group for each sysout file and prevents CSSMTP from holding or deleting sysout files if one sysout file contains an error.

cssmtp1

The external writer name if the ExtWrtName statement is specified in the CSSMTP application configuration file, or the CSSMTP application address space name if the ExtWrtName statement is not specified in the CSSMTP application configuration file.

userid.MYCSSMTP.NOTE

The SMTP commands data set.

SMTP commands

The following are general SMTP commands and data rules for CSSMTP:

Rules:

- The following size limits apply:
 - The SMTP command lines must not exceed 510 characters in length
 - Data lines must not exceed 998 characters in length
 - Path addresses must not exceed 255 characters in length
 - Domain names must not exceed 255 characters in length
 - User names, which are the local part of a mailbox specification, must not exceed 64 characters in length
 - Maximum number of RCPT records per mail message is 2000
- Files must not contain graphic characters (0x00 through 0x3f) except for the tab character (0x09) or newline character (0x15).

Tips:

- Records can be concatenated with the next line when the record is 80 bytes long and the last character is a less than (<) or newline (0x15) character .
- Use the EBCDIC code page. The default code page is EBCDIC IBM 1047. See the [Translate configuration statement](#) information in [z/OS Communications Server: IP Configuration Reference](#) for more details.

Result: Trailing blanks are removed from all records.

| <i>Table 21. SMTP commands that are supported by CSSMTP</i> | | | | |
|---|---------------------------------|--|--|--|
| Subcommand | Supported by SMTP Server | Supported by CSSMTP application | Description | Reference |
| DATA | YES | YES | Defines information as the data text of the mail body. | “DATA command: Define the following information as data” on page 304 |

Table 21. SMTP commands that are supported by CSSMTP (continued)

| Subcommand | Supported by SMTP Server | Supported by CSSMTP application | Description | Reference |
|------------|--------------------------|---------------------------------|--|--|
| EHLO | NO | YES | Identifies the domain name of the sending host to SMTP. | “EHLO command: Identify the domain name of the sending host to SMTP” on page 305 |
| EXPN | YES | NO | Verifies whether a mailbox exists on the local host. | Command is not implemented |
| HELO | YES | YES | Identifies the domain name of the sending host to SMTP. | “HELO command: Identify the domain name of the sending host to SMTP” on page 305 |
| HELP | YES | NO | Provides help with SMTP commands. | Command is not implemented |
| MAIL FROM | YES | YES | Specifies the mail sender. | “MAIL FROM command: Specify the sender of the mail” on page 306 |
| NOOP | YES | NO | Returns a 250 OK return code when SMTP is responding. | Command is not implemented |
| QUEU | YES | NO | Gets information about mail that is queued at SMTP for delivery. | Command is not implemented |
| QUIT | YES | YES | Stops SMTP processing. | “QUIT command: End SMTP processing” on page 306 |
| RCPT TO | YES | YES | Specifies the mail recipients. | “RCPT TO command: Specify the recipients of the mail” on page 307 |
| RSET | YES | YES | Resets the SMTP processing to the initial state. | “RSET command: Reset the SMTP processing to the initial state” on page 307 |
| STARTTLS | NO | YES | Tells the CSSMTP application that the SMTP server is currently able to negotiate the use of TLS. | “STARTTLS command: Indicate the ability to negotiate the use of TLS” on page 307 |
| TICK | YES | NO | Inserts an identifier into the batch SMTP response data set. | Command is not implemented |
| VERB | YES | NO | Enables or disables verbose mode. Note: VERB ON can cause spool problems for SMTP if the REPLY TO: <i>user</i> is not a valid NJE <i>node.user</i> . | Command is not implemented |

Table 21. SMTP commands that are supported by CSSMTP (continued)

| Subcommand | Supported by SMTP Server | Supported by CSSMTP application | Description | Reference |
|------------|--------------------------|---------------------------------|--|----------------------------|
| VRFY | YES | NO | Verifies whether a mailbox exists on the local host. | Command is not implemented |

Note: The EXPN, HELP, NOOP, QUEU, TICK, VERB, and VRFY commands and the optional commands SAML, SEND, SOML, and TURN (see RFC 821 for details), are not implemented in the CSSMTP application. These commands are ignored and are not processed.

Data sets that contain SMTP commands can be written to the JES spool data set as SYSOUT data sets. These SYSOUT data sets contain either punch or NETDATA records. Data sets originate from users on the same system as the CSSMTP address space or from users on any system that is connected to the host system through an NJE network.

DATA command: Define the following information as data

Purpose

Use the DATA command to define the data text that composes the mail body.

Guideline: Use the DATA command after a HELO or EHLO command, a MAIL FROM command, and at least one RCPT TO command have been accepted.

Format

➤ DATA ➤

Parameters

This command has no parameters.

Results:

- If any record in the body of the mail begins with a period (.), the sending SMTP program must convert the period (.) into a pair of periods (..). When the receiving SMTP server encounters a record that begins with two periods in the body of the mail, it discards the leading period. This convention, which must be followed for batch SMTP connections, permits the body of mail to contain records that would otherwise be interpreted as signaling the end of the body of the mail.
- If a QUIT command is not found at the end of a batch SMTP command data set, a QUIT command is implied.
- If the header of the mail in a batch SMTP command is not explicitly specified with the DATE record, the CSSMTP application adds one. You can modify this behavior with the header statement in the configuration. See [Header statement in the z/OS Communications Server: IP Configuration Reference](#).
- If a blank line between the mail header and the mail body is not explicitly specified, the CSSMTP application adds one.
- If a Message-ID header is not explicitly specified in the mail message, the CSSMTP application adds one for this mail message. The Message-ID is a mail message identifier. For example:

```
Message-ID: <TESTMAIL.SYS00006.CSSMTP1.mydomain.com.Sep302008.160454.541437.1>
```

- The Message-ID header consists of the following information:

- TESTMAIL: The job name of the mail message of the JES spool file. This behavior can be modified with the header statement in the configuration. See [Header statement in the z/OS Communications Server: IP Configuration Reference](#).
- SYS00006: The job identifier of the JES spool file. This behavior can be modified with the header statement in the configuration. See [Header statement in the z/OS Communications Server: IP Configuration Reference](#).
- CSSMTP1.mydomain.com: The fully qualified host name on which the CSSMTP application is running
- Sep302008.160454.541437.1: The date and time when the mail message was processed. The value 160454 represents the time 16:04:54.
- If this mail message cannot be delivered, the CSSMTP application appends a U to the end of the Message-ID into the undeliverable mail notification. For example:

```
<TESTMAIL.SYS00006.CSSMTP1.mydomain.com.Sep302008.160454.541437.1U>
```

- If a Message-ID header is explicitly specified in the mail message, the CSSMTP application adds one into the undeliverable mail notification if this mail message cannot be delivered. For example:

```
<TESTMAIL.SYS00006.CSSMTP1.mydomain.com.Sep302008.160454.541437.1U>
```

- If a From header is not explicitly specified, the CSSMTP application adds one with the *sender_path_address* value that is specified on the MAIL FROM command.

Restriction: When mail arrives over the JES spool interface using an SMTP batch or TSO TRANSMIT command, a newline character or less than (<) character in the last column of the 80-byte input record data is interpreted as a continuation character by the SMTP transport layer. The last byte of data should not be a continuation character. If the last byte is a continuation character, then the last record is not processed correctly and data is missed. Check your mail file, correct the data, and resend the mail.

EHLO command: Identify the domain name of the sending host to SMTP

Purpose

Use the EHLO command to identify the domain name of the sending host to SMTP before you issue a MAIL FROM command.

Rule: Send the EHLO command once before a MAIL FROM command.

Requirement: When the STARTTLS command is used, the EHLO command must also be used. Currently, the only mail extension that is supported by the CSSMTP application is the STARTTLS command. See [“STARTTLS command: Indicate the ability to negotiate the use of TLS” on page 307](#) for more information.

Format

➤ EHLO — *domain_name* ➤

Parameters

domain_name

Specifies the domain name of the sending host.

HELO command: Identify the domain name of the sending host to SMTP

Purpose

Use the HELO command to identify the domain name of the sending host to SMTP before you issue a MAIL FROM command.

Format

➤ HELO — *domain_name* ➤

Parameters

domain_name

Specifies the domain name of the sending host.

Usage

- The HELO command is sent once before a MAIL FROM command.

MAIL FROM command: Specify the sender of the mail

Purpose

Use the MAIL FROM command to specify the sender of the mail.

Guideline: Use the MAIL FROM command after a HELO or EHLO command.

Format

➤ MAIL FROM: — <*sender_path_address*> ➤

Parameters

sender_path_address

Specifies the full path address of the sender of the mail.

Requirement: The less than (<) and greater than (>) symbols that surround the *sender_path_address* parameter in the syntax diagram are required.

QUIT command: End SMTP processing

Purpose

Use the QUIT command to end SMTP processing.

Requirement: Ensure that this the last command in the spool file.

Format

➤ QUIT ➤

Parameters

This command has no parameters.

RCPT TO command: Specify the recipients of the mail

Purpose

Use the RCPT TO command to specify mail recipients.

Restrictions:

- A maximum of 2000 RCPT TO commands can be processed on a single note. If you have more than 2000 RCPT commands in your batch job, only the first 2000 RCPT commands are processed; the remainder are ignored.
- You must issue a MAIL FROM command before you can issue an RCPT TO command.

Format

➤ RCPT TO: — <*recipient_path_address*> ➤

Parameters

recipient_path_address

Specifies the full path address of the mail recipient.

Requirement: The less than (<) and greater than (>) symbols that surround the *recipient_path_address* parameter in the syntax diagram are required.

RSET command: Reset the SMTP processing to the initial state

Purpose

Use the RSET command to reset SMTP processing to the initial state. The sender and recipient buffers are erased and the process is ready to begin a new mail transaction.

Format

➤ RSET ➤

Parameters

This command has no parameters.

STARTTLS command: Indicate the ability to negotiate the use of TLS

Purpose

Use the STARTTLS command to tell the SMTP application to negotiate the use of TLS.

Requirement: You must issue an EHLO command before you can issue the STARTTLS command. If you issue the STARTTLS command, it must be issued before you can issue the MAIL FROM command.

When STARTTLS is specified, the scope of the security attribute is set for the mail messages that follow in the JES spool file until another HELO or EHLO SMTP command is issued. See the [CSSMTP security details](#) in [z/OS Communications Server: IP Configuration Guide](#) for more information.

Format

➡ STARTTLS ➡

Parameters

This command has no parameters.

SMTP commands and reply codes across a TCP/IP connection

When SMTP commands arrive over a TCP/IP connection, all responses (positive or negative) are returned over that TCP/IP connection. See RFC 2821 for more detailed information about the SMTP protocol and the SMTP commands and reply code sequences that are generated between an SMTP client and SMTP server.

The SMTP reply code is generated by the remote SMTP server when it receives an SMTP command over a TCP/IP connection. These reply codes might appear in the CSSMTP log file or in error reports. All SMTP reply codes contain a 3-digit number. The first digit of the reply code indicates the successful responses shown in the following table:

| First digit | Description |
|-------------|--|
| 2 | Indicates a positive response; the command was accepted. |
| 3 | Indicates a positive response to the SMTP DATA command. The remote SMTP server is ready to receive the data that is associated with the command. |
| 4 | Indicates a temporary negative response. Reissue this command at a later time. |
| 5 | Indicates a permanent negative response. The command has been rejected. |

CSSMTP exit for unwanted mail

You can use an CSSMTP exit to inspect and filter unwanted mail (spam) that is sent through the CSSMTP application. See [CSSMTP exit](#) information in [z/OS Communications Server: IP Configuration Reference](#) for details.

Example of receiving mail

The CSSMTP application processes the spool file that contains mail messages and forwards those mail messages to a target server where a mail recipient receives the mail. The following is an example of receiving mail:

```
Received: from VIC000.XYZdomain.com (VIC000)
by VIC000.XYZdomain.com (VIC000 [1.1.1.1])
for <USER1.VIC000> (USER1.VIC000)
with ESMTP (IBM CSSMTP z/OS V01R11.00)
Id <USER1C.J0B00035.VIC000@XYZdomain.com.Sep062008.101336.934866.1> ;
Sat, 06 Sep 2008 10:13:37 -0400
Date: Thur, 5 Sep 2008 10:12:57 -0400
From: Carol <CAROL@YOUR.MVSDOMAIN.COM>
To: someuser@some.domain.com
Cc: Alice <alice@ai.our.edu>
Subject: update
Message-ID: <USER1C.J0B00035.VIC000@XYZdomain.com.Sep062008.101336.934866.1>
```

```
Mike: Cindy stubbed her toe. Bobby went to
      baseball camp. Marsha made the cheerleading team.
      Jan got glasses. Peter has an identity crisis.
      Greg made dates with 3 girls and couldn't
      remember their names.
```

Example of an undelivered mail notification

If the CSSMTP application cannot deliver a piece of mail and a report was requested, then the CSSMTP application reports this situation as an undeliverable problem.

If it is requested, an undeliverable mail notification that explains the reason that the mail was not delivered is returned to the original sender. See the [CSSMTP report statement information in z/OS Communications Server: IP Configuration Reference](#) for details. The CSSMTP application might be unable to deliver a mail message because a target server is unreachable or an incorrect recipient user ID on the destination host. If mail cannot be delivered, the body of the original mail message is included in the undeliverable mail notification. The following is an example of an undeliverable mail notification:

```
Mail was not delivered to the following recipients:
<userx@vic000.XYZdomain.com>
Reply text:550 User 'userx' Unknown
```

```
Original mail text:
Received: from vic000.XYZdomain.com (vic000)
by vic000.XYZdomain.com (vic000 [1.1.1.1])
for <USER1.vic000> (USER1.vic000)
with ESMTP (IBM CSSMTP z/OS V01R11.00)
Id <USER1UD2.JOB00046.vic000@XYZdomain.com.Sep072008.132807.37174.1> ;
Sun, 07 Sep 2008 13:28:08 -0400
Subject: test of undeliverable
from: me at vic000
to: user1 at vic000
Date: Sun, 07 Sep 2008 13:28:08 -0400
Message-ID: <USER1UD2.JOB00046.vic000@XYZdomain.com.Sep072008.132807.37174.1>
```

```
Mike: Cindy stubbed her toe. Bobby went to
      baseball camp. Marsha made the cheerleading team.
      Jan got glasses. Peter has an identity crisis.
      Greg made dates with 3 girls and couldn't
      remember their names.
```

Example of generated error reports

When a spool file of SMTP commands is processed, information that is related to detected spool file errors is accumulated in an error file. Errors can be generated because of bad record lengths, commands that are not valid, syntax errors for commands, and other conditions. If any error reports have been generated at completion of the spool file processing, the error report can be sent to the mail administrator or sent to the sysout file. See the [CSSMTP report statement information in z/OS Communications Server: IP Configuration Reference](#) for details.

```
Job USER1UD2/          Error Report for USER1UD2 (JOB00046)
For DDname: SYSUT2      /DEST (JOB00046) created by VIC000.USER1 at Sun, 07 Sep 2008 13:28:07 -0400
Dataset name: USER1.USER1UD2.JOB00046.D0000102.SMTPUDV2
CSSMTP_XYZ generated the following messages:
--- Line    2 Mail      1 : Undeliverable mail for testid@test.com
Message-Id: <USER1UD2.JOB00046.VIC000@XYZdomain.com.Sep072008.132807.37174.1>
Mail was not delivered to the following recipients:
userx@vic000.XYZdomain.com
Reply : 550 User 'userx' Unknown

Completed at Sun, 07 Sep 2008 13:28:08 -0400

1 = mail messages found
0 = mail messages with errors

0 = recipients to whom mail was sent successfully
1 = recipients to whom mail messages could not be delivered

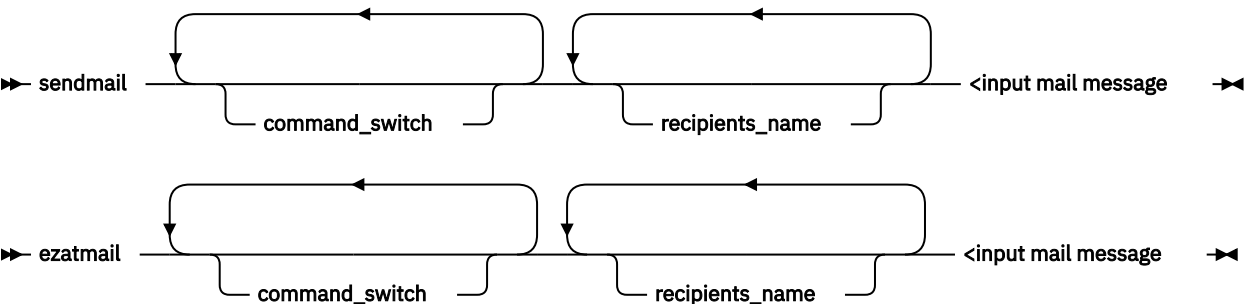
Disposition of the JES file was DELETE
```


Chapter 7. Sending emails by using the sendmail to CSSMTP bridge

Purpose

The sendmail to CSSMTP bridge (sendmail bridge) command allows the user to send emails by using the facilities of the z/OS shell. The command parses input command switches, reads the mail message from the UNIX System Services file, and processes the mail message. The input mail message is updated by adding SMTP commands and SMTP headers, if no headers are specified in the input mail message. The updated mail message is transmitted to the JES spool data set that is processed by the Communications Server SMTP (CSSMTP) application. CSSMTP must be configured and running.

Format



Parameters

Command switches are command-line arguments that begin with a hyphen (-) and precede the list of recipients (if any). The forms for the command-line switches, where -Y is a single letter, are:

- Y Boolean switch
- Yarg or -Y arg Switch with argument

All switches are single letters.

Table 22 on page 311 lists the sendmail bridge command switches.

| Table 22. sendmail bridge command switches | | |
|--|---|--|
| Switches | Description | Argument required after the command switch |
| -bmode | Set operating mode. -bm is the only supported argument. | Yes |
| -bm | Be a mail sender | Yes |
| -Cconfigurationfile or -C configurationfile | Specify configuration file name | Yes |
| -ddebugginglevel | Turn on debug trace. Allow any debugginglevel. | Yes |

Table 22. *sendmail bridge command switches (continued)*

| Switches | Description | Argument required after the command switch |
|---|--|--|
| -F <i>fullname</i> or -F <i>fullname</i> | Set the sender's full name. | Yes |
| -f <i>mailaddress</i> or -f <i>mailaddress</i> | Set the sender's mail address | Yes |
| -i | Treat any line that contains only a single period as ordinary text, not as an EOF indicator. | No |
| -n | Do not do aliasing, alias support is disabled. | No |
| -O | Set an option | Yes |
| -oi | Treat any line that contains only a single period as ordinary text, not as an EOF indicator. | No |
| -t | Get recipients from the mail message header | No |
| -v | Log the content of the built JES spool data set | No |
| -W <i>extwrtrname</i> or W <i>extwrtrname</i> | Set the CSSMTP external writer name | Yes |

Rules:

- If an argument begins with a hyphen (-), it is a command switch; otherwise, it is a recipient.
- An argument after the command can allow for a space between the single command character and the argument. If the argument itself contains the space, the argument must be surrounded by single or double quotes. For example, -F 'firstname lastname'.
- If both "-f" and "-F" are specified, "-F" is ignored.
- The sendmail bridge command can only be invoked as a z/OS UNIX Shell command and through submitting a batch job that invokes BPXBATCH.

Tips:

- For command switches that are not supported by the sendmail bridge, but were supported by sendmail in previous releases, the command switches are ignored. A debug message is logged if the command switch -d is specified. The following is an example of an ignored command: "Ignore - Unsupported command switch '-i'".
- For invalid command switches that sendmail did not support in previous releases, the sendmail bridge command fails with an error logged. The following is an example of an error command: "ERROR: Unknown command switch -Z".

recipients_names

Specifies one or more recipient names to whom the mail should be sent. Each recipient name can be a full mail address or can be an alias name. An alias name does not contain an At sign (@). For more information about the alias name, see sendmail to CSSMTP bridge Configuration Statement O AliasFile option in [z/OS Communications Server: IP Configuration Reference](#).

input mail message

Specifies the z/OS Unix File System full path name that contains one input mail message to be sent. The mail message contains the SMTP header and body but does not contain any SMTP commands.

Detailed description for each command switch:

-bm

Indicate that sendmail bridge is a mail sender. It is an optional switch, and is only included for compatibility with the sendmail client supported in z/OS Communications Server V2R2 and earlier.

-C

Defines the z/OS Unix File System full path for the configuration file name. It is an optional switch, and there is no default path. For more information about the "sendmail to CSSMTP bridge configuration file", see [z/OS Communications Server: IP Configuration Reference](#).

-d

Enables debug mode. When specifying -d, the debugging level is required, but full debugging output is provided regardless of level.

-F

Defines the sender's full name. The argument is used to generate the "From" SMTP header when a "From" SMTP header is not specified in the input mail message; otherwise, the argument is ignored.

-f

Defines the sender's mail address. The argument is used to generate the "MAIL FROM" SMTP command and the "From" SMTP header, if the "From" SMTP header is not specified in the input mail message.

-n

Disables alias search for this send. For more information about sendmail bridge alias support, see [Configuration Statement O AliasFile option in z/OS Communications Server: IP Configuration Reference](#).

-O

Defines the sendmail bridge options:

- OAliasFile
- OMaxAliasRecursion
- OMaxRecipientsPerMessage

The following list shows the search order for finding option parameter:

- The -O command switch is specified
- The O statement

For more information about those options, see [the sendmail bridge Configuration O statement in z/OS Communications Server: IP Configuration Reference](#).

-t

Indicates that the sendmail bridge should get the recipients from the SMTP headers, for example, TO:, CC:, and BCC:, of the input mail message.

-v

Enables tracing by the sendmail bridge of the created JES spool data set that the CSSMTP application will process.

-W

Define the CSSMTP external writer name that the CSSMTP application uses as the selection criteria for the created JES spool data set. For information about CSSMTP configuration statement "ExtWrtName" see [z/OS Communications Server: IP Configuration Reference](#).

The following examples show two different ways to specify the recipients to receive email. In both cases the same recipients will receive the email.

Example1: specify recipients on the command

```
sendmail -d0-99.100 you1@work.com you2@work.com </tmp/mymail1  
/tmp/mymail1 contains:  
From: me@work.com  
Subject: Good job today  
  
Great work!
```

Example2: use -t to get recipients from the input mail message

```
sendmail -d0-99.100 -t </tmp/mymail2  
/tmp/mymail2 contains:  
From: me@work.com  
To: you1@work.com, you2@work.com  
Subject: Good job today  
  
Great work!
```

Chapter 8. Using remote printing

z/OS Communications Server provides client and server support for remote printing. The remote printing application enables you to spool data sets remotely to a Line Printer Daemon (LPD). The Line Printer Requester (LPR) sends the spooled data set to a specified print server host and to a specified printer.

This topic describes the remote printing commands listed in [Table 23 on page 315](#).

Note: Although this information describes the commands and parameters that are valid for the MVS LPR client, you might not get the same results from non-MVS servers, because those servers might not support the same commands and parameters.

Table 23. Remote printing commands

| Command | Description | See |
|----------|--|---|
| LPQ | Request a list of the printer queue on a remote printer. | “LPQ command—Request a list of the printer queue on a remote printer” on page 315 |
| LPR | Print to a remote printer. | “LPR command—Print to a remote printer” on page 317 |
| LPRM | Remove a job from the printer queue on a remote host. | “LPRM command—Remove a job from the printer queue on a remote host” on page 328 |
| LPRSET | Set the default printer and host name. | “LPRSET command—Set the default printer and host name” on page 330 |
| TSO SMSG | Monitor the Status of LPD | “TSO SMSG command—Monitoring the Status of LPD” on page 331 |

LPQ command—Request a list of the printer queue on a remote printer

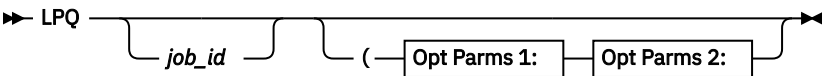
Purpose

Use the LPQ command to request a list of the printer queue on a remote printer from the LPD server controlling that printer.

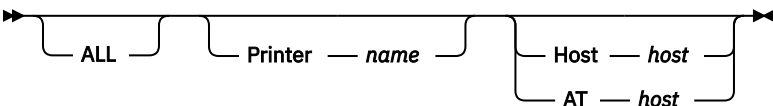
Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPQ command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not active, an ABEND0D6 can occur.

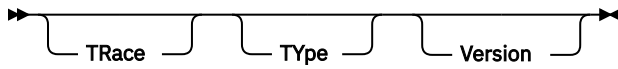
Format



Opt Params 1:



Opt Params 2:



Parameters

job_id

Specifies either a user ID (this cannot start with a digit), or a job number in the remote printer queue. If you do not specify *job_id* with the LPQ command, all the jobs in the remote printer queue are listed.

Note: *job_id* is case sensitive on some systems.

ALL

Prints a long report, which shows the source host and other print job information.

Printer name

Specifies the name of the printer for which you want the printer queue listed.

The printer name cannot contain an @ symbol.

Host host

Specifies the name or IPv4 IP address of the printer host. The name must resolve to an IPv4 address. AT is a synonym for this option.

AT host

Specifies the name or IPv4 IP address of the printer host. The name must resolve to an IPv4 address. Host is a synonym for this parameter.

TRace

Turns on the trace details for interaction with the remote printer. TRace always overrides TYpe because TYpe is a subset of TRace. Use this option whenever you need to document a problem.

TYpe

Displays the progress of the command.

Version

Displays the version of the program.

Examples

- Query the printer lp0 on the system os2sys1 and print a short listing of the jobs that are queued for the lp0 printer:

```
LPQ (PRINTER lp0 HOST os2sys1
```

- If the LPRSET command was previously issued (LPRSET lp0@os2sys1), using the following LPQ command has the same effect as issuing the command in the previous example.

```
LPQ
```

- Get a long listing of the jobs queued, including the name of the host that created the jobs:

```
LPQ (PRINTER lp0 HOST os2sys1 ALL
```

- List the jobs for a user named smith:

```
LPQ smith (PRINTER lp0 HOST os2sys1
```

- Get information only about job 123:

```
LPQ 123 (PRINTER lp0 HOST os2sys1
```

Usage

- If the printer or host name are not specified in the LPQ command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can specify these variables with the LPRSET command. You can use these variables to set up a default printer and host to be used if you do not specify a printer or host.
- User names in a query are case sensitive. For example, smith and SMITH are not the same names. Also for example, on UNIX systems, lp0 and LP0 can refer to different printers.
- Some systems do not answer with the job information when you use a job number for a job that was not produced by the querying system.
- You would not normally use the LPQ command to query an MVS system, because the LPD queue on MVS processes so quickly.
- The input string for parameters is limited to 255 bytes. To use the input string effectively, remove any extra embedded blanks and use shorter parameter labels. For example, use P instead of the fullword Printer as a parameter.

LPR command—Print to a remote printer

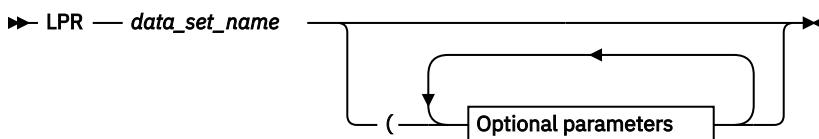
Purpose

Use the LPR command to print to a remote printer.

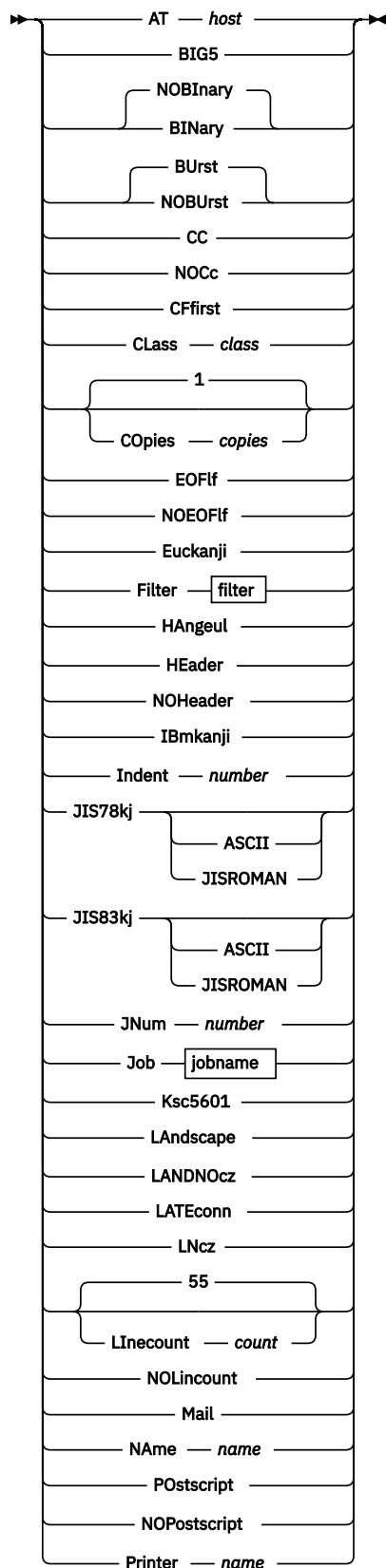
Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPR command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.
3. The TSO LPR command is written in the PASCAL language, so the size of the data set to be printed is limited to 2,147,483,647 (X'7FFFFFFF') bytes.

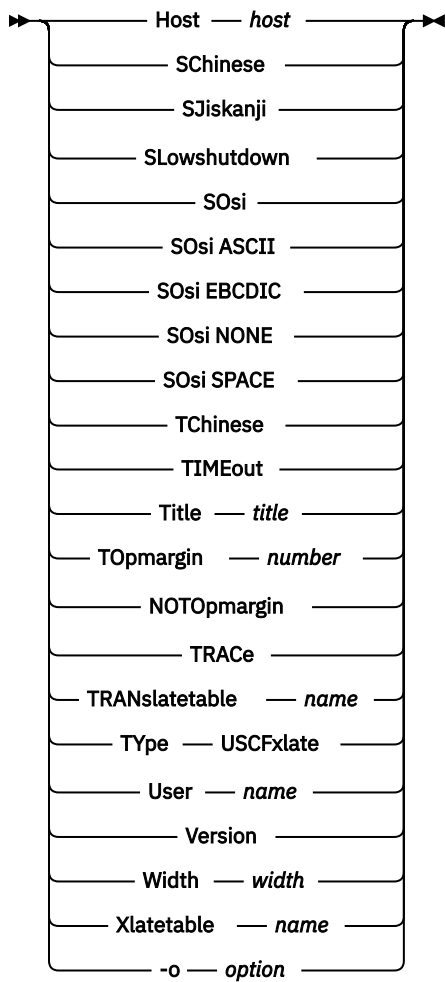
Format



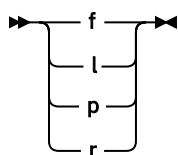
Optional parameters



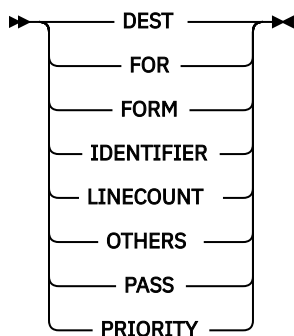
More optional parameters



filter



jobname



Parameters

data_set_name

Specifies the name of the data set to be printed. This cannot be the name of a z/OS UNIX file.

AT *host*

Specifies the name or IPv4 IP address of the printer host machine. If *host* is a name, it must resolve to an IPv4 address. Host is a synonym for this option.

BIG5

Converts data from Traditional Chinese host DBCS to Big-5 DBCS when transferring data to a remote system. LPR loads the BIG5 DBCS translation table from TCPCHBIN binary table data set.

NOBinary

Causes LPR to convert the data from EBCDIC to ASCII when it is sent to the remote system. This is the default.

BINary

Causes LPR to send the data without translation and without any indication of record boundaries. Use this option if the data set is already in ASCII.

Note: The MVS LPD server always converts data sets in ASCII to EBCDIC, and there is no option to disable this conversion.

BURst

Causes a burst (banner) page to be printed on the remote printer. This is the default.

NOBURst

Prevents a burst (banner) page from being printed on the remote printer.

CC

Causes the remote system to interpret the first character of each line as carriage control.

Records containing control characters that are not valid are deleted. If all records in the file are deleted, LPR processes the empty file.

CC is the default if the record format is FA, FBA, FBM, FM, VA, VBA, VBM, or VM. The characters used to specify these record formats have the following meanings:

| Character | Meaning |
|-----------|--|
| F | Fixed record length |
| V | Variable record length |
| B | Blocked records |
| A | Records containing ISO/ANSI control characters |
| M | Records containing machine code control characters |

NOCc

Prevents the remote system from interpreting the first character of each line as an ASA carriage control.

CFfirst

Specifies that LPR will send the ControlFile describing the data before it sends the DataFile that contains the data. This option enables some LPD servers to print larger jobs since the data can be printed as it is received.

Specify this option when small jobs print, but large jobs do not.

Note: Even when this option is specified, the print job might still be too complex to print.

Class *class*

Specifies the class name to the remote system. The class name is printed on the banner pages. The default is the sending host name.

To override the SYSOUT CLASS of a job, the CLASS value must be only a single alphanumeric character . For example, CLASS c.

Copies *copies*

Specifies the number of copies to be printed. The default is one copy.

EOFIf

Causes an ASCII line feed after the last line of data of a file formatted by ANSI carriage control. Since ANSI carriage control positions the paper before the line is printed, you cannot specify how the paper should be positioned after the last line prints. The normal network standard is to terminate every file with a line feed. Some LPDs do not print the last line if the line feed is not added. For example, in ANSI carriage control, a 'F1'X means eject to a new page before printing the lines. ANSI or ASA CC is used in files with RECFM=FBA.

EOFIf is the default for ANSI CC files. Specify NOEOFIf if your paper is positioned incorrectly and an extra line is printed because LPR adds a line feed.

This parameter does not impact files that are not ANSI CC. For example, BINary, POstscript, LANdscape and Machine CC files are not affected by EOFIf.

NOEOFIf

Inhibits adding the ASCII line feed after the last line of a file that is formatted by ANSI carriage control. Since ANSI carriage control positions the paper before the line is printed, how the paper should be positioned after the last line cannot be specified by the carriage control characters. The normal network standard is to terminate every file with a line feed. Specifying this parameter causes the last byte of data to be the last byte sent; a line feed is not added.

For example, an ANSI carriage control 'F1'X means eject to a new page before printing the line. ANSI or ASA CC is used, for example, in files with RECFM=FBA.

Specify NOEOFIf if your paper is positioned wrong because an extra line is being printed at the end of the file. Some applications specify the positioning and do not have the extra line feed sent by LPR.

For example, BINary, POstscript, and Machine CC files are not affected by this parameter. This option does not apply when you specify LANdscape.

Euckanji

Causes the data to be converted from EBCDIC DBCS to Extended UNIX Code kanji ASCII DBCS when it is sent to the remote system. LPR loads the EUCKANJI DBCS translation table from the TCPKJBIN binary translate table data set.

Filter *filter*

Specifies the type of processing to be done on the data by the remote system. The *filter* is written as a single letter. Both uppercase and lowercase letters are accepted, but uppercase letters are converted to lowercase.

Note: The filter values must also be defined in the SERVICE statement of the configuration data set for the MVS LPD server. See the [z/OS Communications Server: IP Configuration Reference](#) for more information on the LPD server configuration data set.

In addition to the following list of filter codes, there are filters supported by other servers described in RFC 1179. [For a list of Requests for Comments (RFCs), see [Appendix D, “Related protocol specifications,”](#) on page 381.]

| Filter code | Description |
|-------------|---|
| f | Print as a sequence of lines |
| l | Print, passing through all control characters |
| p | Print with pagination |

| Filter code | Description |
|-------------|---|
| r | <p>Print, interpreting the first column as FORTRAN carriage control characters. The supported IBM FORTRAN carriage control characters are 1, 0, +, and -.</p> <p>When using the MVS LPD server to print files with filter code r, the PAGESIZE parameter on the SERVICE statement for the printer you are printing to also affects pagination.</p> <p>See “Usage” on page 327 for further information.</p> |

HAngeul

Causes the data to be converted from EBCDIC DBCS to Hangeul ASCII DBCS when it is sent to the remote system. LPR loads the HANGEUL DBCS translation table from the TCPHGBIN binary translate table data set.

HHeader

Causes a page header to be inserted by the client at the top of every printed page if the NOCc and NOBINARY options are in effect. To cause the server to insert page headers, use **p** as the value of *filter* and specify the NOHeader option.

NOHeader

Prevents the client from inserting page headers.

IBmkanji

Causes the data to be sent without translation as IBM (EBCDIC) kanji. This parameter performs the same function as the BINARY parameter.

Indent *number*

Specifies the number of columns the remote system indents the output when **f** or **p** is specified as the value of *filter*.

JIS78kj ASCII

Causes the data to be converted from EBCDIC DBCS to JIS 1978 kanji ASCII DBCS, using the ASCII shift-in escape sequence ESC (B, when it is sent to the remote system. LPR loads the JIS78KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS78kj JISROMAN

Causes the data to be converted from EBCDIC DBCS to JIS 1978 kanji ASCII DBCS, using the JISROMAN shift-in escape sequence ESC (J, when it is sent to the remote system. LPR loads the JIS78KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS83kj ASCII

Causes the data to be converted from EBCDIC DBCS to JIS 1983 kanji ASCII DBCS, using the ASCII shift-in escape sequence ESC (B, when it is sent to the remote system. LPR loads the JIS83KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS83kj JISROMAN

Causes the data to be converted from EBCDIC DBCS to JIS 1983 kanji ASCII DBCS, using the JISROMAN shift-in escape sequence ESC (J, when it is sent to the remote system. LPR loads the JIS83KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JNum *number*

Specifies a specific job number for the print request, where *number* is a unique, 3-digit number in the range 000 - 999. This job number is used by LPR to name the temporary data and control files, such as:

```
dFA123hostname  cFA123hostname
```

The JNum parameter is not valid from NPF; do not specify JNum in the NPF OPTIONS file.

If JNum is not specified for LPR, the three-digit number is randomly generated by LPR.

Job *jobname*

Specifies the job name to the remote system. The default name is the full data set name. The job name is printed on the banner pages.

The following parameters are available with the Job parameter and are entered as `PARAMETER=value`. When you use these parameters, you must separate them from Job or *jobname* by a blank and from other options by a comma and no blank. For example,

```
Job PASS=password,F0R=userid
```

DEST

Sets the destination node. The default is the node on which the LPR client is running.

FOR

Specifies a user ID other than the sending user ID for which the output is to be spooled. The default is the sender's ID.

FORM

Identifies the form on which the data is printed. This is the equivalent of the form-name subparameter in the MVS SYSOUT parameter on the DD card. Therefore, the form name specified can be 1 through 4 alphanumeric or national (\$, #, @) characters.

IDENTIFIER

Sets the destination ID. The default is SYSTEM.

LINECOUNT

Specifies a numeric field indicating the number of lines on a page. This option overrides the PAGESIZE parameter of LPD.

OTHERS

Causes all subsequent options to be ignored. This option is ignored by the MVS LPD server.

PASS

Specifies the password. The default is no password, which causes the job to fail if the RACF option is specified for the service.

PRIORITY

Specifies the transmission priority. The default is 50.

Ksc5601

Causes the data to be converted from EBCDIC DBCS to Korean Standard Code KSC-5601 ASCII DBCS when it is sent to the remote system. LPR loads the KSC5601 DBCS translation table from the TCPHGBIN binary translate table data set.

Landscap

Converts a non-PostScript data set to a PostScript data set for printing with print lines parallel to the long edge of the paper. If the remote printer can process PostScript output, the data set is printed in landscape format (rotated 90 degrees). Some nonprinting EBCDIC characters below X'3F' are changed to blanks.

Note: An ASCII cntl-Z X'1A' is sent after the data. If this dos-EOF character causes problems, use the LANDNOcz option instead.

LANDNOcz/LNcz

Converts a non-PostScript data set to a PostScript data set for printing with print lines parallel to the long edge of the paper. If the remote printer can process PostScript output, the data set is printed in landscape format (rotated 90 degrees). Some nonprinting EBCDIC characters below X'3F' are changed to blanks.

Note: Normally, you should use the Landscap option, unless the dos-EOF character (X'1A') sent after the file causes problems. LN and LNcz are abbreviations for LANDNOcz.

LATEconn

Causes LPR to process the input data file before making any TCPIP connection with the printer. When this option is not specified, the TCPIP connection is made before the data file processing begins. This option can be specified when the initial processing of very large files causes the connection with a printer to be dropped because of a timeout.

LInecount count

Determines the number of lines to be printed before a new heading is printed. This option is meaningful only for a data set that does not have the CC option specified either explicitly or by default.

The valid range for LInecount is 0 to the Pascal integer MAX number 2 147 483 647. The default value is 55. To suppress printing a header before each new page, specify LInecount 0. Specifying LInecount 0 has the same effect as specifying the NOLinecount option.

NOLinecount

Prevents a header being printed before each new page. Specifying NOLinecount has the same effect as specifying LInecount 0.

Mail

Causes mail to be sent to the user when the printing operation ends (for those servers that support this).

NAmE name

Specifies the job information to be provided by the remote system in response to a query. Only the *name* or *files* portion of the query is displayed. This option is not honored by all remote printing servers.

POstscript

Inserts the header required by some systems to recognize a PostScript data set.

NOPostscript

Prevents a PostScript data set from being recognized as a PostScript data set.

Printer name

Specifies the name of the printer on which you want the data set printed. The printer name is case sensitive.

Host host

Specifies the name or IPv4 IP address of the printer host machine. If *host* is a name, it must resolve to an IPv4 address. AT is a synonym for this option.

SChinese

Converts data from Simplified Chinese host DBCS to Simplified Chinese PC DBCS when transferring data to a remote system. LPR loads the SCHINESE DBCS translation table from TCPSCBIN binary table data set.

SJiskanji

Causes the data to be converted from EBCDIC DBCS to Shift JIS kanji ASCII DBCS when it is sent to the remote system. LPR loads the SJISKANJI DBCS translation table from the TCPKJBIN binary translate table data set.

SLOWshutdown

Causes slower TCP/IP connection termination after the job is sent to the printer. This option is rarely needed. This option is provided for print servers which discard the print job just successfully received when LPR uses the fast shutdown. Before using this option, check the print server for errors (such as spool file full) that might be causing the job to be discarded instead of printed.

SOsi

Determines how any EBCDIC DBCS shift-out ('OE'X) and shift-in ('OF'X) characters in the input file are handled. The ASCII, EBCDIC, or SPACE parameters specifies what is used as shift characters in the ASCII output from LPR. If SOsi is specified without a following parameter, ASCII is used as the default.

If you do not specify SOsi, shift-out/shift-in characters are not used in the ASCII data stream. Therefore, the EBCDIC DBCS shift characters are just removed during the translation to ASCII. This is the same as specifying SOsi NONE.

SOsi has no effect on DBCS translations JIS78KJ, JIS83KJ, and IBMKANJI. It is used with other DBCS translation such as BIG5, EUCKANJI, HANGEUL, KSC5601, SCHINESE, SJISKANJI, and TCHINESE.

SOsi ASCII

Specifies that DBCS data strings in the ASCII output are delimited by special shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out ('0E'X) becomes ('1E'X), and '0F'X becomes '1F'X.

SOsi EBCDIC

Specifies that DBCS data strings in the ASCII output are delimited by EBCDIC shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out ('0E'X) remains EBCDIC shift-out ('0E'X), and '0F'X remains '0F'X.

SOsi NONE

Specifies that DBCS data strings in the ASCII output are not delimited by any shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, any EBCDIC DBCS shift characters are removed.

SOsi SPACE

Specifies that DBCS data strings in the ASCII output are delimited by ASCII space ('20'X) characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out ('0E'X) becomes ASCII space ('20'X) and EBCDIC shift-in ('0F'X) also becomes '20'X.

TChinese

Causes the data to be converted from EBCDIC DBCS to Traditional Chinese (5550) ASCII DBCS when it is sent to the remote system. LPR loads the TCHINESE DBCS translation table from the TCPCHBIN binary translate table data set.

TIMEout

Specifies that LPR wait 5 minutes for an ACK or NACK from the LPD printer. If ACK or NACK does not arrive, LPR terminates the connection with Error Number=73. Since LPR waits for ACK in several places, this error can occur in different LPR messages, such as EZB1048E.

For some printers, the ACK is not returned until the job has printed. For these printers, specifying the TIMEout option can cause long jobs to stop printing.

When the TIMEout option is not specified, LPR waits as long as the TCP/IP connection exists. For some printers, this could tie up the PORT (and any NPF thread using LPR on that PORT) until an operator intervenes to fix the printer. For NPF users, TIMEout should be added as an LPR option for these printers. See the [z/OS Communications Server: IP Network Print Facility](#) for information about configuring LPR options in NPF.

Title *title*

Specifies the title assigned to a data set printed with the FILTER p option.

TOPmargin *number*

Specifies the number of lines designated for the top margin.

NOTOPmargin

Indicates that blank lines are not inserted at the top of each page.

TRACe

Turns on the trace details for interaction with the remote printer. TRACe always overrides TYPe because TYPe is a subset of TRACe.

TRANslatetable *name*

Specifies the SBCS translate table to be used by the client. The *name* parameter is preceded by either the *userid* or the *hlq* and followed by TCPXLBIN to form the data set name of the translate table (*userid.name.TCPXLBIN* or *hlq.name.TCPXLBIN*). If both data sets exist, a search order hierarchy determines which one is to be used.

See the [z/OS Communications Server: IP Configuration Reference](#) for more information about search order hierarchy, loading, and customizing SBCS translation tables. XLatetable is a synonym for this option.

TYPe

Displays the progress of the command as the data set is being processed.

USCFxlate

Specifies that a single byte translation table such as JPNKANA be used for the print data. The control file generated by LPR and sent to LPD contains upper- and lowercase alphanumeric characters. Specify this option if any of them are being translated incorrectly, causing the LPD to reject the print jobs.

User name

Specifies a name that overrides the user identification of the program that is requesting the print job, prints on the banner page, and becomes the user identification of the mail option. The *name* field cannot be longer than eight characters. If you do not enter the user name parameter, it defaults to the system user identification or to the job name.

Version

Displays the version of the program.

Width width

Specifies the line width of a data set printed with the FILTER options *f*, *l*, *p*, or *r*.

Xlatetable name

Specifies the SBCS translate table to be used by the client. The *name* parameter is preceded by either the *userid* or the *hlq* and followed by TCPXLBIN to form the data set name of the translate table (*userid.name.TCPXLBIN* or *hlq.name.TCPXLBIN*). If both data sets exist, a search order hierarchy determines which one is to be used.

See the [z/OS Communications Server: IP Configuration Reference](#) for more information about search order hierarchy, loading, and customizing of SBCS translation tables. TRANslatetable is a synonym for this option.

-o option

Specifies an option that the control file in PSF for AIX (PSF/6000) or InfoPrint uses to format the print job. Any -o option honored by PSF/6000 can be passed to the control file using this parameter of the LPR command. This parameter must be issued without a blank between the -o and the option.

Below are some sample options. None of these options are standard LDP options. They are not honored by most LPD print servers.

-ochars=GT15

Sets the value of the CHARS parameter to GT15.

-obin=2

Sets the input bin to 2 (use alternate input bin).

-opagedef=P13700

Sets the value of PAGEDEF parameter to P13700.

-oformdef=F1SEPA

Sets the value of the FORMDEF parameter to F1SEPA.

-ocopies=002

Sets the JCL COPIES count to 2. This parameter is not honored by most LPD servers. Other LPD servers ignore all control file information and print one copy. For these printers, issue the LPR command multiple times to get multiple copies.

IBM Print Services for AIX (S544-3878-03) contains detailed descriptions of the -o options for PSF/6000.

Examples

- Print the data set TEST.LISTING on a printer named lp0 on the system mvs1:

```
LPR TEST.LISTING (PRINTER lp0 HOST mvs1
```


- If TEST.LISTING has a record format that contains carriage control such as VBA, the first character of each line is interpreted as carriage control. To prevent the first character of each line from being interpreted as carriage control, use the following command:

```
LPR TEST.LISTING (PRINTER lp0 HOST mvs1 NOCC
```

- If this LPRSET command was issued:

```
LPRSET lp0@mvs1
```

the following LPR command would also print the data set on printer LP0 on the host MVS1 and prevent the first character of each line from being interpreted as carriage control:

```
LPR TEST.LISTING (NOCC
```

- Print the data set TEST.LISTING in landscape mode:

```
LPR TEST.LISTING (LANDSCAPE
```

The following example shows the PostScript attributes used in the Landscape option:

```
614 25 translate 90 rotate .88 .76 scale
/n 1 def
/fs 10 def
/ls 11.2 def
/ld ls 2 mul def
/lt ls 3 mul def
/t 740 fs sub def
/y t def /ff t def /os 20 def
/s 512 string def
/Courier-Bold findfont
fs scalefont setfont
/p {n {copypage} repeat erasepage} def
/i (%stdin) (r) file def
/{c i read not {p stop} if def
c 26 eq {p stop} if
/x 20 def
/y c 43 eq {y /x os def}
{c 32 eq {y ls sub}
{c 48 eq {y ld sub}
{c 45 eq {y lt sub}
{c 49 eq {ff} {y} ifelse}
ifelse} ifelse} ifelse} ifelse def
/ff 0 def
y 65 le {p /y t def} if x y moveto
/os i s readline not {p stop} if dup show
length 0 eq {20} {20.72} ifelse def } loop
```

To understand these attributes you might need to reference a PostScript manual.

- If a data set TEST has a low-level qualifier of LISTPS (PostScript), use the following command to send TEST to a PostScript-capable printer without specifying the PostScript option:

```
LPR TEST.LISTPS (PRINTER lp0 HOST mvs1
```

- Print a FORTRAN source program with 57 lines on each page:

```
LPR TEST.FORTRAN (LINECOUNT 57
```

Usage

- The input string for parameters is limited to 255 bytes. To use the input string effectively, remove any extra embedded blanks and use shorter parameter labels. For example, use P instead of the full word Printer as a parameter.

- When sending a print job to a printer that has RACF in its definition in the LPD.CONFIG data set, you must specify the password. If it is for a different user ID, you must specify that password and user ID as follows:

```
Job PASS=password,FOR=userid
```

- If the printer or host name are not specified in the LPR command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can specify these variables with the LPRSET command.
- LPR normally issues messages only if there is an error. If you want to track the progress of the command, use the TYPe or TRACe parameter.
- You can use the LPR command to send PostScript data sets to a printer that can print documents in that language. LPR checks that no incompatible options were given, if it is a PostScript data set. You can override this check, if you want to print a PostScript program with the NOPostscript option. UNIX systems examine the first few characters of a data set (looking for %!) to determine if a data set is a PostScript data set. If you have PostScript data sets that do not contain the characters %!, use the POstscript parameter to add them.
- Carriage control is interpreted line by line. A data set can mix ASA and machine carriage control. Interpretation is done by converting the controls to the appropriate ASCII sequences, before the data set is sent to the remote system. Lines that have incorrect carriage control are not printed.
- When a data set is printed without carriage control, LPR adds a heading line that shows the name of the data set, the title of the system on which the LPR command is running, and a page number. You can specify the number of lines to be printed (excluding the 3 heading lines) with the LInecount parameter.
- When you specify a filter code, LPR ignores CC, HEader, NOCc, NOHeader, and TOPmargin. When a filter code of f, l, p, or r is specified, LPR stops paginating the data set it is printing. Instead, it sends the data in the data set as plain lines. The following list provides a description of these filter codes:

| Filter code | Description |
|-------------|--|
| f | Print as a sequence of lines. |
| l | Print, passing through all control characters. |
| p | Print with pagination. |
| r | Print, interpreting the first column as FORTRAN carriage control characters. The supported IBM FORTRAN carriage control characters are 1, 0, +, and -. |

When using the MVS LPD server to print files with filter code r, the PAGESIZE parameter on the SERVICE statement for that printer is ignored in the LPD CONFIG file. The PAGESIZE parameter defaults to 60 if it is not specified. To prevent unwanted page ejects, change the value specified on the PAGESIZE parameter to a number greater than the actual number of lines on the pages being printed. For example, you can specify 100 000 for the PAGESIZE parameter.

When you specify a filter code of c, d, g, n, t, or v, LPR transmits the data as a byte stream (as though you specified the BINary option).

LPRM command—Remove a job from the printer queue on a remote host

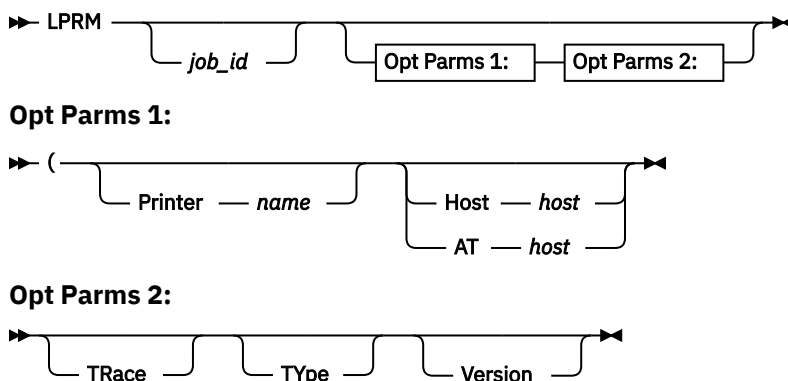
Purpose

Use the LPRM command to remove a job from the printer queue on a remote host.

Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPRM command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format



Parameters

job_id

Specifies either a user ID (this must not start with a digit), or a job number in the remote printer queue. If you do not specify *job_id* with the LPRM command, your currently active job is removed.

Printer name

Specifies the name of the printer associated with the job.

Host *host*

Specifies the name or IPv4 IP address of the printer host. If *host* is a name, it must resolve to an IPv4 address. AT is accepted as a synonym for HOST.

AT *host*

Specifies the name or IPv4 IP address of the printer host. If *host* is a name, it must resolve to an IPv4 address. Host is a synonym for this option.

TRace

Turns on the trace details for interaction with the remote printer. TRace always overrides TYpe because TYpe is a subset of TRace.

TYpe

Displays the progress of the command.

Version

Displays the version of the program.

Examples

- Cancel job number 123 on the printer lp0 on the local system os2sys1:

```
LPRM 123 (PRINTER lp0 HOST os2sys1
LPRM 123 (PRINTER lp0 AT os2sys1
```

If the job is in the queue, it is removed. If the job is currently active, it is stopped.

- If the LPRSET command was previously issued (LPRSET lp0@os2sys1), using the following LPRM command has the same effect as issuing the command in the previous example:

```
LPRM 123
```

- Cancel the currently active job:

```
LPRM (PRINTER lp0 HOST os2sys1
```

Usage

- The input string for parameters is limited to 255 bytes. To use the input string effectively, remove any extra embedded blanks and use shorter parameter labels. For example, use P instead of the full word Printer as a parameter.
- If the printer and host name are not specified in the LPRM command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can set these variables with the LPRSET command. You can use these variables to set up a default printer, which is used if you do not specify a printer.
- Removing the currently active job can depend on the number of jobs currently printing. If you have two jobs printing, and you use the LPRM command without the *job_id* parameter, the first job might finish, but you could inadvertently remove the second job instead.

LPRSET command—Set the default printer and host name

Purpose

Use the LPRSET command to set the default printer and host name. The printer and host name can also be included in the line printer commands (LPR, LPQ, and LPRM).

Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPRSET command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format

➤ LPRSET — *printer@host* — Optional Parameters:

Optional Parameters:

➤ (Query TRace Version TYpe)

Parameters

printer@host

Specifies the name of the printer and host to be used. If *host* is a name, it must resolve to an IPv4 address.

Query

Displays the current settings for the default printer and host.

TRace

Turns on the trace details for the recording of the printer and remote host name. TRace always overrides TYpe because TYpe is a subset of TRace.

TYpe

Displays the progress of the command.

Version

Displays the version of the program.

Examples

- Set the default printer and host as the printer lp0 on the local system mvs1:

```
LPRSET lp0@mvs1
```

- Display the current version of LPRSET:

```
LPRSET (VERSION
```

- Display the current settings:

```
LPRSET (QUERY
```

Usage

- The input string for parameters is limited to 255 bytes. To use the input string effectively, remove any extra embedded blanks and use shorter parameter labels. For example, use TR instead of the fullword Trace as a parameter.
- When you use LPRSET to set the printer and host, a data set by the name *userid.LASTING.GLOBALV* (where *userid* is the TSO user ID) is created or updated. If this data set does not exist, you must be able to create this data set. *userid.LASTING.GLOBALV* is the name required by LPRSET.
- Printer names can be case sensitive. The printer name must be spelled the way the host uses it. For example, on UNIX systems, lp0 and LP0 can refer to different printers.

Also be aware that ISPF panels default to uppercase unless otherwise specified.

- When you query the current settings on the same command that you set the default printer and host, the query of the current settings is done before the specified default printer and host are set. For example, if the printer was set to PRINT1@RALVM13, and you perform the query, LPRSET PRINT2@RALVM13 (Q, the message returned says, PRINTER is PRINT1@RALVM13. However, it also changes the default printer to PRINT2. So if you query again, it would respond, PRINTER is PRINT2@RALVM13.

If you want to set the default printer and host and see that the defaults were set in the same LPRSET command, use the TYPE parameter. For example, if you want to set the default printer and host to PRINT2 on RALVM13 and then see the results afterward, you would enter the following command.

```
LPRSET PRINT2@RALVM13 (TYPE
```

TSO SMSG command—Monitoring the Status of LPD

Purpose

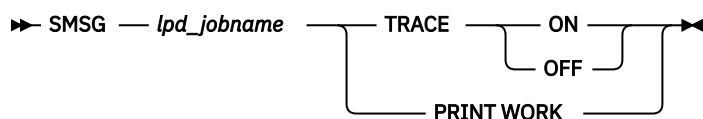
The TSO SMSG command provides an interactive interface to the LPD server to:

- Turn on and off diagnostics tracing
- Query the work queue currently being used by LPD server

These commands are privileged so the commands are accepted only from users specified in the OBEY statement in the LPD server configuration data set. For more information regarding this statement, see the [z/OS Communications Server: IP Configuration Reference](#). Responses to the SMSG commands are sent to the SYSPRINT file that is associated with the LPD started procedure job.

Note: The TSO SMSG LPD command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not active, the following message is issued to the terminal of the TSO user: EZY2040I SMSG: VMCF is not active on the system.

Format



Parameters

TRACE ON

Enables tracing in the LPD server.

TRACE OFF

Disables tracing in the LPD server.

PRINT WORK

Prints the jobs in the work queue for LPD.

Examples

The following example shows the output from the SMSG LPD PRINT WORK command, which is sent to the SYSPRINT file.

Note: Response to the SMSG LPD command does not display on the screen for the user. The user must look in the SYSPRINT file that is associated with the LPD job to view the response.

```
16:59:29 EZB0786I Command received "PRINT WORK".
16:59:29 EZB0731I      Work Queue start
16:59:29 EZB0732I      <job number>      <job state>
16:59:29 EZB0733I      Work Queue end
```

Chapter 9. Using GDDMXD/MVS with the X Window System

This topic describes GDDMXD/MVS and the GDDMXD CLIST. This information also describes how to use GDDMXD/MVS user-specified options and keyboard functions. Problem determination information associated with GDDMXD/MVS is also described in [z/OS Communications Server: IP Diagnosis Guide](#).

Note: The feature HIP612X is required for GDDMXD.

The following subjects are covered in this topic:

- [“Overview of GDDMXD/MVS” on page 333](#)
- [“Using GDDMXD/MVS” on page 335](#)
- [“GDDMXD/MVS: User-specified options” on page 336](#)
- [“GDDMXD keyboard functions” on page 344](#)

Overview of GDDMXD/MVS

GDDMXD/MVS is an interface that allows graphics from the IBM Graphical Data Display Manager/MVS to be displayed on workstations that support the X Window System. When GDDMXD/MVS is installed and activated, the data stream created by the GDDM application is translated to the X Window System protocol and transmitted by TCP/IP to the X Window System server for the display. If GDDMXD/MVS is installed and not activated, or has been made inactive, the GDDM application transmits data to its supported display device as if GDDMXD/MVS were not present.

GDDMXD/MVS keyboard and character set mappings

The following member names contain the described character sets:

| Member name | Description |
|-------------|---|
| GDXALTCS | A member of SEZAINST that contains a second character set (the 3270 alternate character sets). |
| GDXAPLCS | A member in the SEZAINST data set that contains sample keyboard mapping for APL2. |
| KEYCODE | A member of SEZALOAD that displays key codes. You can edit this data set to change keyboard mappings. |

GDDM: Executable code

The following member names contain the described executable code or sample:

| Member name | Description |
|-------------|--|
| GDXLIOX0 | A member of SEZALOAD that contains executable code |
| XDEFAULT | A member of SEZAINST that contains the sample XDEFAULT member for GDDMXD/MVS |

GDDM application limitations

When GDDMXD/MVS is inactive, there are no GDDM application restrictions. The following types of functions are not supported by GDDMXD/MVS:

- Multiple instances of the GDDM application

- Opening multiple display devices at one time
- Operator windows

GDDM display limitations

GDDMXD/MVS appears as an IBM 3179G device model to the GDDM application. When the HostRast option is active, the device model is IBM 3279. The IBM 3179 Model G Color Graphics Display Station is used regardless of the display device nickname presented by the application.

The following are characteristics of the GDDMXD/MVS IBM 3179G display station:

Alphanumeric Cursor

When the graphics cursor is not enabled, the alphanumeric cursor can be repositioned by pointing the X Window System pointer device cursor for the GDDMXD window at the wanted character location and pressing the pointing device button.

Attached Graphics Cursor

When the graphics cursor is attached, the X Window System pointer device cursor for the GDDMXD window changes to a crosshair pattern and moves as the pointer device is moved.

Blinking Character Attribute

Ignores the blinking character attribute.

Character Display

Characters with an EBCDIC value of less than hex 40 are displayed as blanks.

Color Mixing

GDDMXD/MVS supports only the overpaint foreground color mix mode. The initial color of the image area is black, and mixing with the actual background colors is not performed.

An exception is made for data passed by an image data order. In this exception, a combined foreground color mix mode is supported, if the series of begin image orders have exactly the same parameter values.

When the HostRast option is active, color mixing is performed by GDDM, and the preceding exception does not apply.

Default Vector Symbol Set

The Default Vector Symbol Set is not supported.

Detached Graphics Cursor

When the graphics cursor is detached by the data stream or is not attached, the X Window System pointer device cursor for the GDDMXD window changes to an open arrow when the keyboard is unlocked, or changes to an X shape when the keyboard is locked.

Detectable Fields

Ignores the detectable fields.

Pixel Spacing

When the HostRast option is not active, the vertical and horizontal pixel spacing of the actual display device that is obtained from the X Window System is supplied to the GDDM application. When the HostRast option is active, the pixel spacing of an IBM 3279 Color Display Station is supplied to the GDDM application.

Visual Appearance

For all programmed symbol and image data that is received from the GDDM application, each GDDM pixel is mapped to one X Window System display pixel, which causes a different appearance from the same data stream displayed on an IBM 3179G Color Graphics Display Station. This map process can also cause display differences in the placement of alphanumeric field data over the graphics display and in the appearance of the filled areas of the graphic display. When the HostRast option is active, aspect ratio distortion of the displayed graphics appears, unless the aspect ratio of the X Window System display is the same as the IBM 3279.

Using GDDMXD/MVS

Before GDDM data can be sent to an X Window System display, activate GDDMXD/MVS by invoking the GDDMXD CLIST. Make sure that you have already copied the GDDMXD CLIST to your system CLIST data set. If you have not, see the [z/OS Communications Server: IP Configuration Reference](#) for instructions.

Note: If you do not want to run GDDM applications through the X Window System, do not enable GDDMXD/MVS.

To invoke the GDDMXD CLIST, use the GDDMXD command in the format described in [“GDDMXD command—Invoke the GDDMXD CLIST”](#) on page 335.

GDDMXD command—Invoke the GDDMXD CLIST

Purpose

Use the GDDMXD command to invoke the GDDMXD CLIST.

Format

➔ GDDMXD — ON —➔
 └ OFF ┘

Parameters

ON

Enables GDDMXD/MVS. GDDM output is sent to the X Window System display. The system responds with GDDMXD/MVS active.

OFF

Disables GDDMXD/MVS. The system erases the data set that was created when GDDMXD/MVS was activated and responds with GDDMXD/MVS inactive.

Identifying the target display

Purpose

A TSO global variable is used by the X Window System to identify the IP address of the target display based on the contents of the data set *user_id.XWINDOWS.DISPLAY*.

Format

➔ *internet_address* — *:target_server* — *.target_screen* —➔

Parameters

internet_address

Specifies the IPv4 IP address of the host machine on which the X Window System server is running.

:target_server

Specifies the number of the display server on the host machine.

.target_screen

Specifies the screen to be used on the *target_server*.

Examples

Examples of the contents of the *user_id.XWINDOWS.DISPLAY* data set.

- `charm.cambridg.ibm.com:0.0`
- `129.42.3.109:0.0`

GDDMXD usage notes

- When you use the TSO Session Manager with GDDMXD, enter a null line in the host session window after the GDDMXD graphics window disappears. This updates and activates the host session window.
- When you run PL/I GDDM applications, do not let the ISASIZE run-time option default to 0. This causes excessive allocation of storage below the 16 megabyte line, and causes a variety of storage allocation abends. Enter a run-time option for ISASIZE, such as `ISASIZE(20K)`, to prevent storage allocation abends.
- Although GDDMXD provides its own device information to the GDDM application, normal GDDM device initialization occurs. A full-screen 3270 TSO session from a real or emulated 3270 terminal with 80 columns and 32 rows is required to invoke the GDDM application.

Resizing the GDDMXD graphics window

GDDMXD supports four graphics presentation space sizes. The size of the graphics presentation space used by GDDMXD is determined by the window width specified by the Geometry option in the `user_id.X.DEFAULTS` data set (see “[GDDMXD/MVS: User-specified options](#)” on page 336 for more information). The size is also determined dynamically when you resize the GDDMXD graphics window. The relationship between the size of the graphics presentation space and the window width is shown in [Table 24 on page 336](#).

Table 24. Supported graphics presentation space sizes

| Window width (pixels) | GDDMXD graphics presentation space (pixels) |
|-----------------------|---|
| < 650 | 480 horizontal by 352 vertical |
| >=650 to < 850 | 720 horizontal by 512 vertical |
| >= 850 to <= 1024 | 960 horizontal by 682 vertical |
| > 1024 | 1200 horizontal by 864 vertical |

For graphics presentation space sizes other than the default size (720 pixels by 512 pixels), bit-mapped data, such as symbol sets and images, is expanded or compressed to meet the scaling requirements of the specified graphics presentation space.

You can expand bit-mapped data by duplicating rows and columns of the data. The resulting view can differ slightly from the default-size view. You can compress single-plane bit-mapped data by combining rows and columns of the data with a logical OR function. Because this might not yield acceptable results when a black on white image is viewed, the `Compr` option is provided to specify that a logical AND function be used to compress the data. See “[Compr option—Control the technique used to compress bit-mapped data](#)” on page 339 for more information about using this option. You can compress multiplane bit-mapped data by eliminating certain rows and columns. Data compression produces a view that is different from the default-size view.

GDDMXD/MVS: User-specified options

The user-specified options for GDDMXD/MVS are entries in a data set called `user_id.X.DEFAULTS`. The `user_id.X.DEFAULTS` data set is searched during initialization of GDDMXD/MVS.

Note: The values in the `user_id.X.DEFAULTS` data set are case sensitive and must be entered as shown.

The options listed in [Table 25 on page 337](#) are supported by GDDMXD/MVS.

Table 25. GDDMXD/MVS options

| Option | Description | See |
|----------|--|---|
| ANFontn | Specifies the X Window System font used for characters in the alphanumeric presentation space. | “ANFontn option—Specify the X Window System font used for characters in the alphanumeric presentation space ” on page 337 |
| CMap | Specifies whether the default color map is loaded or bypassed. | “CMap option—Specify whether the default color map is loaded or bypassed ” on page 338 |
| Compr | Controls the technique used to compress bit-mapped data when a graphics window size of 480 by 352 pixels is specified. | “Compr option—Control the technique used to compress bit-mapped data ” on page 339 |
| Enter | Overrides the default key mapping for Enter. | “Enter option—Override the default key mapping for Enter ” on page 339 |
| GColornn | Specifies a color name. | “GColornn option—Specify a color name ” on page 339 |
| Geometry | Specifies the size and location of the initial GDDMXD graphics presentation space. | “Geometry option—Specify the size and location of the initial GDDMXD graphics presentation space ” on page 341 |
| GMCPnn | Overrides GDDM multicolor patterns with workstation color names. | “GMCPnn option—Override GDDM multicolor patterns with workstation color names ” on page 341 |
| HOSTRAST | Performs raster image processing at the host. | “HostRast option—Perform raster image processing at the System/370 host ” on page 342 |
| NewLine | Overrides the default key mapping for NewLine. | “NewLine option—Override the default key mapping for NewLine ” on page 342 |
| XSync | Requests that the X Window System process one request at a time. | “XSync option—Request that the X Window System process one request at a time ” on page 343 |
| ZWL | Tells GDDMXD/MVS to draw all lines using 0-width lines. | “ZWL option—Tell GDDMXD/MVS to draw all lines using 0-width lines ” on page 343 |

ANFontn option—Specify the X Window System font used for characters in the alphanumeric presentation space

Purpose

Use the ANFontn option to specify the X Window System font that GDDMXD should use to display characters in the alphanumeric presentation space of the GDDMXD window.

Format

➤ gddmx*ANFont — *n* — : — *fontname* ➤

Parameters

n

Specifies presentation space size.

fontname

Specifies the name of the X Window System font.

Examples

The following are examples of ANFontn options.

```
gddmx*ANFont1: Rom8
gddmx*ANFont3: Rom14
```

Usage

Graphics mode 1 and 2 characters in the graphics presentation space are not affected by this option. The value of *n* is in the range 1 - 4 and defines the X Window System font for each of the four sizes of presentation space supported by GDDMXD. You can specify the ANFontn option for any, all, or none of the four values for *n*. The X Window System fonts specified should be fixed-space fonts that have characters that fit into the character box size required by each of the four presentation space sizes.

| <i>n</i> | Presentation space | Character box | Example font |
|----------|--------------------|---------------|--------------|
| 1 | 480 x 352 | 6 x 11 | Rom8 |
| 2 | 720 x 512 | 9 x 16 | Rom11 |
| 3 | 960 x 682 | 12 x 21 | Rom14 |
| 4 | 1200 x 864 | 15 x 27 | Rom17 |

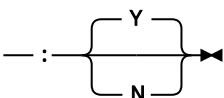
If you select a font that has characters that are larger than the character box size, the characters might overlap when displayed.

CMap option—Specify whether the default color map is loaded or bypassed

Purpose

Use the CMap option to specify whether the default color map is loaded or bypassed.

Format

➔ gddmx*CMap — : 

Parameters

Y

Directs GDDMXD/MVS to load the default color map. This is the default.

N

Directs GDDMXD/MVS to bypass loading the default color map.

Usage

During initialization, GDDMXD/MVS issues the X Window System call, XInstallColormap, to load the default color map. If the CMap option is specified as N, the XInstallColormap call is not made. This option is for X Window System servers that load their own color map and do not want the clients to load any other color map.

Compr option—Control the technique used to compress bit-mapped data

Purpose

Use the Compr option to control the technique used to compress bit-mapped data when a graphics window size of 480 by 352 pixels is specified.

Format

➤ gddmx*Compr — : { O A } ➤

Parameters

O or o

Specifies that a logical OR function must be used when compressing bit-mapped data. This is the default.

A or a

Specifies that a logical AND function must be used when compressing bit-mapped data.

Enter option—Override the default key mapping for Enter

Purpose

The Enter option can be specified in the *user_id*.X.DEFAULTS data set to identify which X Window System Keysym is to be mapped to the Enter function. This option overrides the default mapping of the Keysym XK_Execute to the Enter function.

Format

➤ gddmx*Enter — : — *keysym_name* ➤

Parameters

keysym_name

Specifies the X Window System Keysym representing the physical key. For standard Keysyms, the XK_ prefix is not included in specifying the option.

Examples

In the following example of the Enter option, the X Window System Keysym, XK_Return, is mapped to the Enter function.

```
gddmx*Enter: Return
```

GColornn option—Specify a color name

Purpose

GDDMXD/MVS provides a default mapping of GDDM colors to X Window System colors. Use the GColornn option to override a default color name or to specify a color if a default color name is not available by your X Window System server.

Format

➤ `gddmx*GColor` — *nn* — : — *c* ➤

Parameters

***nn*:**

Specifies the GDDM color entry that is mapped.

c

Specifies the X Window System color that is used as the GDDM color.

Usage

Table 26 on page 340 lists the GDDM colors that GDDMXD/MVS maps to the X Window System.

Table 26. GColors

| GColor <i>nn</i> | GDDM color | X Window System color |
|------------------|----------------|-----------------------|
| GColor1 | Blue | Blue |
| GColor2 | Red | Red |
| GColor3 | Magenta | Magenta |
| GColor4 | Green | Green |
| GColor5 | Turquoise | Turquoise |
| GColor6 | Yellow | Yellow |
| GColor7 | White | White |
| GColor8 | Black | Black |
| GColor9 | Dark Blue | Dark Slate Blue |
| GColor10 | Orange | Orange |
| GColor11 | Purple | Plum |
| GColor12 | Dark Green | Dark Green |
| GColor13 | Dark Turquoise | Dark Turquoise |
| GColor14 | Mustard | Wheat |
| GColor15 | Gray | Gray |
| GColor16 | Brown | Brown |

Examples

The following is an example of using a GColor*nn* option to override a default color:

```
gddmx*GColor3: Pink
```

In this example, specifying the GColor13 entry in the *user_id*.X.DEFAULTS data set maps the GDDM color, magenta, to the X Window System new color of pink.

Geometry option—Specify the size and location of the initial GDDMXD graphics presentation space

Purpose

Use the Geometry option to specify the size and location of the initial GDDMXD graphics presentation space.

Format

➤ `gddmx*Geometry` — : — *width* — * — *height* — + — *x_offset* — + — *y_offset* ➤

Parameters

width

Specifies the initial width of the GDDMXD graphics window. The *width* determines the initial size of the graphics presentation space.

height

Specifies the initial height of the GDDMXD graphics window.

x_offset

Specifies the location of the upper left corner of the window where *x_offset* is the horizontal offset from the upper left corner of the display.

y_offset

Specifies the location of the upper left corner of the window where *y_offset* is the vertical offset from the upper left corner of the display.

Examples

The following is an example of a Geometry option:

```
gddmx*Geometry: 750x600+20+20
```

GMCPnn option—Override GDDM multicolor patterns with workstation color names

Purpose

Use the GMCPnn option to override GDDM multicolor patterns with workstation color names.

Format

➤ `gddmx*GMCP` — *nn* — : — *c* ➤

Parameters

nn:

Specifies the GDDM multicolor pattern.

c

Specifies the color that is used with the defined GDDM multicolor pattern.

Examples

The following is an example of a GMCPnn option:

In this example, the color medium blue is used when multicolor pattern 126 is specified by the GDDM application.

HostRast option—Perform raster image processing at the System/370 host

Purpose

Use the HostRast option to perform raster image processing at the System/370 host.

Use the HostRast option when:

- Multiplane character symbol sets are required by the application.
- GDDM color mixing is important to the application.

The default device model for GDDMXD/MVS is an IBM 3179G with a mouse and the raster image processing is performed at the workstation.

Format



Parameters

N

Directs GDDMXD/MVS to use the IBM 3179G as a device model. This is the default.

Y

Directs GDDMXD/MVS to use the IBM 3279 as a device model.

X

Directs GDDMXD/MVS to use the IBM 3279 as a device model and expand the pixel mapping to reduce aspect ratio distortion.

Usage

- The APL2 character set is not supported when the HostRast option is active.
- When the HostRast option is specified as Y, the GDDM application performs the raster image processing and transmits the picture as a series of characters whose pixel definitions have been transmitted to Programmed Symbol Sets. The picture is mapped exactly as an IBM 3279.
- If the ratio of horizontal to vertical pixel spacing is not the same as that of an IBM 3279, the aspect ratio can be distorted.

NewLine option—Override the default key mapping for NewLine

Purpose

The NewLine option can be specified in the *user_id.X.DEFAULTS* data set to identify which X Window System Keysym is to be mapped to the NewLine function. This option overrides the default mapping of the Keysym XK_Return to the NewLine function.

Format

➤ gddmx*NewLine — :— *keysym_name* ➤

Parameters

keysym_name

Specifies the X Window System Keysym representing the physical key. For standard Keysyms, the XK_ prefix is not included in specifying the option.

Examples

In the following example of the NewLine option, the X Window System Keysym, KP_Enter, is mapped to the NewLine function:

```
gddmx*NewLine: KP_Enter
```

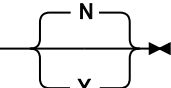
XSync option—Request that the X Window System process one request at a time

Purpose

The X Window System operates asynchronously. By the time an error has been detected, more requests could have been issued by the application.

Use the XSync option to request that the X Window System process one request at a time.

Format

➤ gddmx*XSync:  ➤

Parameters

N

Allows the X Window System to operate asynchronously. This is the default.

Y

Directs GDDMXD/MVS to cause the X Window System to operate synchronously.

Usage

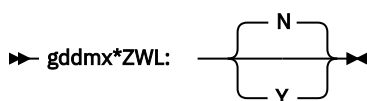
Be aware that system performance goes down when you use XSync=Y.

ZWL option—Tell GDDMXD/MVS to draw all lines using 0-width lines

Purpose

The X Window System supports a range of line widths. Because some X Window System servers draw wide lines at a slow rate, you can use the Zero Width Lines (ZWL) to tell GDDMXD/MVS to draw all lines using 0-width lines. The X Window System server uses the fastest process to draw the lines. The resulting line might not be exactly the same as if it had been drawn as a wide line.

Format



Parameters

N

Directs GDDMXD/MVS not to use 0-width lines for all drawing. This is the default.

Y

Directs GDDMXD/MVS to use 0-width lines for all drawing.

GDDMXD keyboard functions

The following sections detail different keyboard functions supported by GDDMXD/MVS.

GDDMXD/MVS keyboard functions

When you enter input to the GDDM application by GDDMXD/MVS, use the following 3270 keyboard functions.

- All alphanumeric keys
- **F1 - F24**
 - If **F13 - F24** are not available, use **Shift + F1** to **Shift + F12**
- **Tab** or **Shift + Tab**
- Directional arrows
- **End** key to erase to the end of the field
- **Insert** key and **Delete** key
- **PA1, PA2, and PA3**
- **Enter** key
- **Newline** key

Note: The **Backspace** key is treated as a cursor left key.

If you cannot locate these keys on your workstation, see your workstation X Window System documentation to determine the mapping of X Window System key symbol definitions to the physical keys.

GDDMXD/MVS to X Window System keyboard functions

The following are the GDDMXD/MVS keyboard functions that translate to X Window System key symbol definitions. Key functions not listed are not supported.

| GDDMXD/MVS keyboard function | X Window System key symbol |
|------------------------------|----------------------------------|
| APL2 character set toggle | XK_Backspace with state Mod1Mask |
| Clear | XK_Pause |
| Delete | XK_Delete |
| Down | XK_Down |
| End | XK_End |
| Enter | XK_Execute |

| GDDMXD/MVS keyboard function | X Window System key symbol |
|------------------------------|----------------------------|
| Insert | XK_Insert |
| F1 – F12 | XK_F1 – XK_F12 |
| Left | XK_Left |
| Newline | XK_Return |
| PA1 | XK_Prior |
| PA2 | XK_Next |
| PA3 | XK_Home |
| Right | XK_Right |
| Tab | XK_Tab |
| Up | XK_Up |

APL2 character set keyboard

The APL2 character set is activated by simultaneously pressing the X Window System XK_Backspace key (usually the **Backspace** key) and the State Mod1Mask key (usually the **Alt** key). For example, if you use the IBM 101 Enhanced Keyboard, the APL2 character set is toggled on and off by pressing and holding the **Alt** key, and then pressing the **Backspace** key.

When the APL2 character set is active, the characters *APL* are displayed in the title bar of the GDDMXD/MVS window.

In the X Window System, a key code is assigned to each key on the keyboard. GDDMXD/MVS uses key codes in combination with modifier keys. For example, the **Shift** and **Alt** keys determine the data that should be passed back from GDDMXD/MVS to the X Window System application to identify the user's keystroke data.

GDDMXD looks for the data set SEZAINST(GDXALTCS) when it is initialized. Before using GDDMXD, copy the installed TCP/IP copy of SEZAINST(GDXALTCS) to *user_id*.GDXALTCS.PSS, or allocate the common installed copy of GDXALTCS.PSS to ddname GDXDACSP.

A default map for the APL2 character set is provided in GDDMXD/MVS, which corresponds to the IBM 101 Key Enhanced Keyboard. You can override this default map by creating a data set called GDXAPLCS.MAP to define the map for your workstation. When GDDMXD/MVS is initialized, the system searches for a data set called GDXAPLCS.MAP. If the GDXAPLCS.MAP data set exists, the data in the GDXAPLCS.MAP data set replaces the default mapping for all keys.

Setting up *hlq*.GDXAPLCS.MAP

The GDXAPLCS.MAP data set is created to override the default map. It is used to define the map for the workstation. The following steps describe how to set up the GDXAPLCS.MAP data set.

Procedure

1. Invoke the program KEYCODE from the TCP/IP load module library in SEZALOAD to determine the key codes for the keyboard keys.

When KEYCODE is executed from your workstation session to the host system, the key code is displayed for each key pressed at the workstation. Therefore, you can establish the association between a key and the character you want to generate.

See Appendix B, "Mapping values for the APL2 character set," on page 365 for more information about the mapping values that are defined in the GDXAPLCS.MAP data set.

2. Copy the *hlq*.GDXAPLCS.SAMPMAP installed with TCP/IP to *hlq*.GDXAPLCS.MAP.

3. Edit GDXAPLCS.MAP to establish the association between the key codes in the program KEYCODE and the character set and code values in [Appendix B, “Mapping values for the APL2 character set,” on page 365](#).
4. GDDMXD looks for the data set *user_id*.GDXDAPLCS.MAP when it is initialized. If you want to use a different data set name, allocate the data set to ddname GDXDACSM.

Chapter 10. Executing commands on a remote host

The Remote Execution Protocol (REXEC) and the Remote Shell Protocol (RSH) are remote execution clients that enable you to execute a command on a remote host and receive the results on the local host. You can execute either REXEC or RSH from the TSO command line, the z/OS UNIX command line, or as a batch program. When executed as a batch program, the results are stored in a data set for later use.

To use REXEC, you must have a REXEC daemon running on the remote host. The REXEC client passes the user name, password, and command to the REXEC daemon. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

To use RSH, you must have an RSH daemon running on the remote host. The RSH client passes the local user name, remote user name, and command to the RSH daemon. The remote user name can be in the form user/password when the RSH daemon is on an MVS host. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

This information describes how to use the REXEC and RSH clients.

The following subjects are covered in this topic:

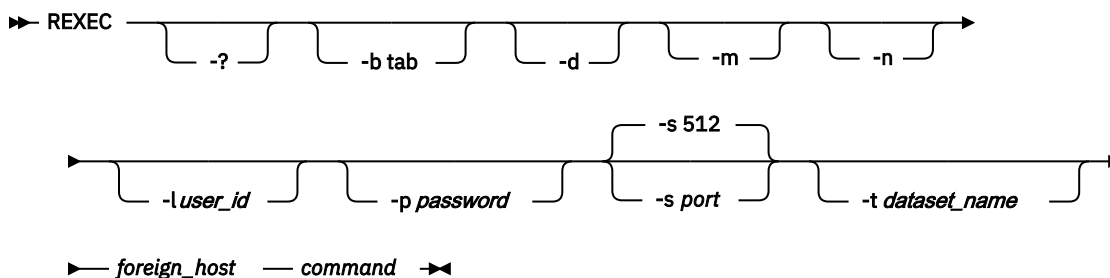
- [“REXEC command - Execute a command on the remote host and receive the results on your local host” on page 347](#)
- [“Using the NETRC data set” on page 349](#)
- [“Submitting REXEC and RSH requests in batch” on page 350](#)
- [“Using remote execution clients in a z/OS UNIX environment” on page 355](#)
- [“The z/OS UNIX orexec/rexec command—Execute a command on the remote host” on page 355](#)

REXEC command - Execute a command on the remote host and receive the results on your local host

Purpose

Use the REXEC command to execute a command on the remote host and receive the results on the local host.

Format



Restrictions:

- The REXEC parameters `-b`, `-d`, `-l`, `-m`, `-n`, `-p`, `-s`, and `-t` are case sensitive and must be entered in lowercase letters. The `user_id` and `password` parameters might be case sensitive, depending on the operating system of the remote host.
- The total length of all specified options cannot exceed 2048 characters. If more characters are specified, the command that is sent to the server will be truncated, with unpredictable results.

Parameters

-?

Displays the help message.

-b *tab*

Specifies the Tab setting. Valid values are in the range 1-12 and the default value is 1.

-d

Activates debug tracing.

-m

Specifies that the machine control character (X'09') is added to the beginning of the output lines for the data sets that are associated with the SYSPRINT or OUTPUT DD cards and have the machine control attribute. When you use this parameter, it should be the first parameter that is passed to REXEC so that all output lines are changed.

-n

Prevents use of the NETRC data set.

-l *user_id*

Specifies the user ID on the foreign host.

-p *password*

Specifies the password for the user ID on the foreign host.

-s *port*

Specifies the TCP port number of the REXEC server on the foreign host. The default, 512, is the port number defined in /etc/services.

-t *dataset_name*

Specifies the data set name of a translation table to be used. The search order when the -t parameter is specified:

```
userid.tran_table.TCPXLBIN  
hlq.tran_table.TCPXLBIN
```

If this data set is not found, REXEC terminates with message EZA4805I.

Note: If the -t parameter is not specified on the invocation of the REXEC command, a hardcoded default table is used that is identical to the STANDARD member in the SEZATCPX data set.

foreign_host

Specifies the name or IP address of the foreign host to which you are sending the REXEC command. Specify the foreign host by its host name or IP address. When you are using IPv6 link-local addresses, you can provide scope information along with the host name or IP address, as described in the [support for scope information in the z/OS Communications Server: IPv6 Network and Application Design Guide](#).

command

Specifies the command to be executed on the remote host.

Example

Use the REXEC command without using NETRC.DATA:

```
READY  rexec -l user28 -p user28 -s 512 mvs1 lista  
  
MVS TCP/IP REXEC CS V1R2  
SYS1.HELP  
GIM.SGIMCLS0  
DSN230.DSNCLIST  
USER.CLIST  
BUILD.CLIST  
SYS1.HRFCLST  
USER28.RSHD5.JOB00160.D0000103.?
```

Usage

- Omitting the **-p** *password* causes the NETRC data set to be searched by following the search order.
- The **-n** parameter can be used to suppress usage of the NETRC data set.
- If the *user_id*, or the *password*, or both are not provided as input parameters or specified from usage of the NETRC data set, the system prompts you to supply the parameters.
- When you issue a command that is to be executed on the remote host, do not place the command in quotation marks. Doing so can result in unexpected results.

Note: There is no such restriction when using the z/OS UNIX `orexec` command.

- The condition code 12 is set when an REXEC batch request encounters one of the following error conditions:
 - The client cannot connect to TCP/IP.
 - The host name cannot be resolved.
 - The translation table cannot be loaded.
- When REXEC issues a command to a remote system and retrieves the output for presentation to the user, it expects a line of output to be no more than 32767 bytes. A warning message is issued when the size of the output line approaches this threshold. This condition has no effect on the job running on the remote host.
- The Tab setting is used exclusively by the client and is not forwarded to the server. The tab setting value determines the alignment of the output. If a Tab character is included in the output data stream, blank characters are included in the output up to the next Tab setting. The processing is the same whether the output is displayed or written to a data set.

Using the NETRC data set

The NETRC data set provides you with an alternative to specifying the *user_id* and *password* values as REXEC parameters. REXEC uses the following search order to find the NETRC data set to use:

1. NETRC DD statement
2. *userid*.NETRC.DATA
3. *tso_prefix*.NETRC
4. *userid*.NETRC

For information about using the NETRC data set in a batch file, see [“Submitting REXEC and RSH requests in batch”](#) on page 350.

If the *password* is specified on the **-p** parameter on the REXEC command, no NETRC data sets are used. The keywords *machine*, *login*, and *password* must be specified in lowercase. The user ID and *password* might be case sensitive and if supplied in the incorrect case, failures might occur when connecting to a REXEC server. Contact the administrator of the server's system if you are uncertain which case should be used.

The *hostname* value that is specified after the *machine* keyword can include scope information, as described in the [support for scope information](#) in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

Guideline: If either *hostname%scope* or *IPv6_address%scope* is specified as the *foreign_host* command parameter, there must be an entry in the NETRC data set that contains a string identical to the value that follows the *machine* keyword so that the correct user ID and *password* values are selected.

The following is the format of the NETRC data set:

```
machine hostname login user_id password password
```

Note: You can omit your *password* in the NETRC data set. If you do, the REXEC command prompts you for your current *password*.

The following is an example of a NETRC data set:

```
machine mvs1 login user28 password abcdef
```

The following is a sample of the response that displays after using the REXEC command and the NETRC data set.

```
READY  rexec mvs1 lista
MVS TCP/IP REXEC CS V1R2
SYS1.HELP
GIM.SGIMCLS0
DSN230.DSNCLIST
USER.CLIST
BUILD.CLIST
SYS1.HRFCLST
USER28.RSHD5.JOB00161.D0000103.?
```

Submitting REXEC and RSH requests in batch

You usually run REXEC and RSH interactively by entering the command and then receiving the results at your terminal. However, you can also run REXEC and RSH as a batch job. To accomplish this, you must supply the necessary job control language (JCL) and submit it to the job entry subsystem (JES) using the TSO SUBMIT command.

The command format when submitted as a batch job is the same as the command format described in [“RExec command - Execute a command on the remote host and receive the results on your local host”](#) on page 347. You enter the command as a parameter on the EXEC statement. The results of the command executed on the remote host are stored on the local host according to how you define the SYSPRINT DD statement. The data set characteristics should be consistent with the output from the command you are executing at the remote host.

When you invoke the REXEC command, a check is made to see if a data set is allocated to INPUT. If a data set is allocated, any input is read from that data set rather than from your terminal. Similarly, a check is made to see if data set is allocated to OUTPUT. If so, all REXEC output is written to that data set rather than to your terminal.

The REXEC or RSH server does not support output being returned to output DD statements other than SYSTSPRT, SYSPRINT or OUTPUT. Sometimes it is necessary to direct the output from a REXEC or RSH batch request to an OUTPUT DD and to include a SYSPRINT DD with SYSOUT=* specified. This is particularly true if the REXEC or RSH command to be executed contains a slash(/).

There might be times when REXEC and RSH are being executed and the user does not want the output to be directed to the SYSPRINT DD file. If a //OUTPUT DD card is coded, output will be directed to it. This DD statement must be directed to a SYSOUT or a validly defined data set. If multiple REXEC or RSH commands are being executed in one step, then it is recommended that DISP=MOD be coded on the //OUTPUT DD card. This will allow each command execution to be appended to the previous output. If the //OUTPUT DD card specifies a nullfile (ie. DSN=NULLFILE or DD DUMMY), then the //OUTPUT DD statement *will not* be used. Regardless of which ddname you use for the REXEC or RSH output, a new data set must have the DCB=(...) parameter specified on the DD card. Any data set you use must have a logical record length (LRECL) greater than 0. If the LRECL is equal to 0 then the data set is not used.

When using the REXEC server, the procedure specified in the TSOPROC argument of the startup procedure must have the //SYSTSPRT DD statement appearing before any other output DD specifications in the procedure. For example, if the batch procedure specified was TSOPROC=TESTJOB, the following example would be the correct specification for the batch procedure for REXEC:

```
//TESTJOB EXEC PGM=IKJEFT01,REGION=4M,DYNAM=30,REGION=4M
//STEPLIB DD DSN=A.LOADLIB,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

Adding a PARM argument to the EXEC JCL DD statement for commands to be submitted to batch might cause unpredictable output, characters to be lost, or output to be presented out of order.

If DDs are dynamically allocated, the order of output returned might be unpredictable. For example, the following TSO/E CLIST were invoked as the remote execution command, using the procedure TESTJOB:

```
PROC 0
TIME
ALLOC FI(OUT) SYSOUT(*)
OPENFILE OUT OUTPUT
SET &OUT = &STR(THIS; IS THE FIRST LINE)
PUTFILE OUT
SET &OUT = &STR(THIS; IS THE SECOND LINE)
PUTFILE OUT
TIME
SET &OUT = &STR(THIS; IS THE LAST LINE)
PUTFILE OUT
CLOSEFILE OUT
FREE FI(OUT)
EXIT
```

In this case, the output might be returned as:

```
THIS IS THE FIRST LINE
THIS IS THE SECOND LINE
THIS IS THE LAST LINE
output from time
output from time
```

In the following example, the TESTJOB procedure was modified to add the OUT DD statement:

```
//TESTJOB EXEC PGM=IKJEFT01,REGION=4M,DYNAM=30,REGION=4M
//STEPLIB DD DSN=A.LOADLIB,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//OUT DD SYSOUT=*
```

The CLIST was also modified, as in the following example (notice that the ALLOC FI(OUT) and the FREE FI(OUT) have been removed):

```
PROC 0
TIME
OPENFILE OUT OUTPUT
SET &OUT = &STR(THIS; IS THE FIRST LINE)
PUTFILE OUT
SET &OUT = &STR(THIS; IS THE SECOND LINE)
PUTFILE OUT
TIME
SET &OUT = &STR(THIS; IS THE LAST LINE)
PUTFILE OUT
CLOSEFILE OUT
EXIT
```

The output appears as in the following example:

```
output from time
output from time
THIS IS THE FIRST LINE
THIS IS THE SECOND LINE
THIS IS THE LAST LINE
```

Be aware that output being returned to the client from DDs other than SYSTSPRT or SYSPRINT might have characters truncated. This behavior has been observed only in the first line of new output files other than SYSTSPRT or SYSPRINT. If you use DDs other than SYSTSPRT or SYSPRINT, you might want to ensure that the first line of the output file contains a blank line, so that no data is lost.

Notes:

1. You can also use the NETRC data set name described in [“Using the NETRC data set”](#) on page 349 to specify the user ID and password. You can override the NETRC data set search order by specifying a NETRC DD statement in the batch job. The NETRC DD statement identifies the NETRC data set to be used. You must provide all REXEC command information by using the NETRC data set and the PARM keyword on the EXEC statement.

2. Submitting a long running command can cause the REXEC program to end abnormally with a 522 system abend code. This can be avoided by specifying TIME=1440 on the EXEC statement of the JCL you submit. Job step timing is suppressed, including the collection of SMF job time accounting information.
3. If the command to be executed on the remote host contains a slash (/), you must use a preceding slash (/) in the input stream of the NETRC data set or the PARM.

The following example shows invoking the RSH program in batch and executing the command:

```
LS ./bin/temp/*
//RSHBATCH EXEC PGM=RSH,
//          PARM='/-l userid hostname ls ./bin/temp/*'
```

4. A condition code of 1 will be set when an REXEC batch request encounters one of the following error conditions:
 - The client cannot connect to TCP/IP.
 - The host name cannot be resolved.
 - The translation table cannot be loaded.
5. A condition code of 1 will be set when an RSH batch request encounters an error condition in which the client cannot connect to TCP/IP or when the host name cannot be resolved.

The following example shows REXEC JCL Spooling Output to JES:

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//          PARM='-l userid -p password foreign_host command'
//SYSPRINT DD  SYSOUT=*
```

Note: The data set containing the JCL cannot have sequence numbers.

The following example shows the use of the *userid*.NETRC.DATA containing the user ID and password. The output is sent to a permanent data set for later use.

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//          PARM='foreign_host command'
//SYSPRINT DD  DSN=USERID.REXEC.SYSPRINT,DISP=(NEW,CATLG),
//          UNIT=3380,VOL=SER=MYVOL
```

Note: When running REXEC in batch, the user ID assigned to the job is used as the *user_id* in the NETRC data set.

The following example shows the use of the NETRC DD statement in batch. The NETRC DD statement can be used in batch to override the default *userid*.NETRC.DATA or *userid*.NETRC files.

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//          PARM='foreign_host command'
//SYSPRINT DD  SYSOUT=*
//NETRC    DD  DSN=TST.REXEC.NETRC,DISP=SHR
```

Note: The user ID and password are retrieved from TST.REXEC.NETRC instead of *userid*.NETRC.DATA or *userid*.NETRC.

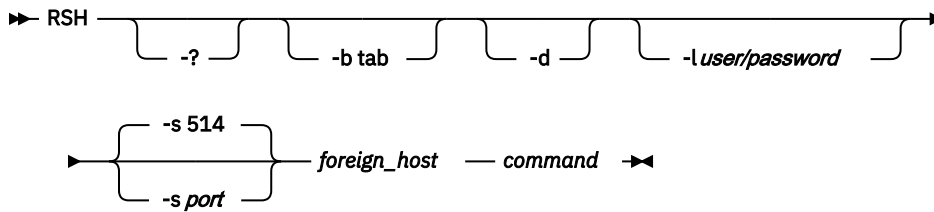
RSH command—Execute a command on a remote host and receive the results on your local host

Purpose

The Remote Shell Protocol (RSH) is a remote execution client similar to REXEC that enables you to execute a command on a remote host and receive the results on your local host.

You can use the RSH command from TSO or as a batch job.

Format



Notes:

1. The parameters for the RSH command could be case sensitive, depending on the remote host you are targeting.
2. The RSH command uses the MESSAGECASE setting in the TCPIP.DATA file to determine whether messages are issued in uppercase or mixed case characters.

Parameters

-?

Lists the valid parameters for the RSH command.

-b tab

Specifies the Tab setting. Valid values are in the range 1 - 12. The default value is 1.

-d

Turns on debug tracing.

-l user/password

Specifies the remote user ID and password.

You must enter the slash (/) between the user ID and password if the target system is MVS. The character you must use can vary, depending on the target host. For VM hosts, you should use an @ character.

-s port

Specifies the server port. The default is 514.

foreign_host

Specifies the name or IP address of the remote host on which you are issuing the command. When you are using IPv6 link-local addresses, you can provide scope information along with the name or IP address, as described in the [support for scope information](#) in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

command

Specifies the command to be executed on the remote host.

Examples

- Execute a command on a remote host and receive the results on your local host:

```

READY

rsh -l user28/user28 mvs1 lista

SYS1.HELP
BUILD.HELP
GIM.SGIMCLS0
ISR.V3R5M0.ISRCLIB
DSN230.DSNCLIST
DSN230.NEW.DSNCLIST
DSN230.DSNAMACS
USER.CLIST
BUILD.CLIST
SYS1.HRFCLST
ISP.V3R5M0.ISPEXEC

READY

```

- Use JCL to submit the RSH command as a batch job:

```

//USER28 JOB ,CARTER,MSGLEVEL=(1,1),NOTIFY=USER28
//RSH31 EXEC PGM=RSH,
//          REGION=800K,
//          PARM='/-d -l user28/user28 mvs1 lista'
//SYSPRINT DD SYSOUT=*
//SYSTCPD DD DSN=USER28.TCPIP.DATA,DISP=SHR
//SYSIN DD DUMMY

```

Usage

- You must enter the required parameters on the command line. The RSH command does not prompt you for missing parameters, or enable you to use the NETRC.DATA set.
- A condition code 1 is set when an RSH batch request encounters an error condition in which the client cannot connect to TCP/IP or when the host name cannot be resolved.
- When RSH issues a command to a remote system and retrieves the output for presentation to the user, it expects a line of output to be no more than 32767 bytes. A warning message is issued when the size of the output line approaches this threshold. This condition has no effect on the job running on the remote host.
- The tab setting is used exclusively by the client and is not forwarded to the server. The tab setting value determines the alignment of the output. If a tab character is included in the output data stream, blank characters are included in the output up to the next tab setting. The processing is the same whether the output is displayed or written to a data set.

RHOSTS.DATA data set

The *user_id*.RHOSTS.DATA data set provides you with an alternative to specifying RSH parameters *user_id* and *password* when you invoke the RSH command.

The *user_id*.RHOSTS.DATA data set contains one or more entries. Each entry consists of two parts, a fully qualified name of a local host and a local userid associated with that local host. The local userid is case sensitive.

The user ID specified in *user_id*.RHOSTS.DATA can be either the user ID you would otherwise specify as an RSH parameter or your logon ID on your local host.

If your user ID is the same at both the local and remote hosts, use this common ID to create RHOSTS.DATA. In this case, you do not need to include the *user_id/password* parameter on the RSH command, as shown in the following example:

```
rsh mvsone lista
```

If your user ID at the remote host is different from your user ID at the local host, use the user ID of the remote host to create RHOSTS.DATA. In this case, you can invoke the RSH command without the password, as follows:

```
rsh -l user28 mvsone lista
```

If you do not create the RHOSTS.DATA data set on the remote host, you must specify both the user ID and the password with the RSH command, as shown in the following example:

```
rsh -l user28/abcdef mvsone lista
```

The host names in the following examples are the official, fully qualified names of local hosts from which you want to run RSH. The user IDs are the logon IDs for those local hosts. Nicknames are not allowed.

```
local.host.name user_id  
mvsthree.raleigh.ibm.com user30  
mvsthree.raleigh.ibm.com user31  
mvstfour.raleigh.ibm.com user30
```

Tip: The RSH server code obtains the remote host name by resolving the remote IP address into a host name. If the remote host connects using a link-local address, then the remote host name that is generated by the resolver can have the format *hostname%scope*. Adding scope information to the appropriate RHOSTS.DATA remote host definitions results in a more efficient search for a matching remote host name. See the [support for scope information in the z/OS Communications Server: IPv6 Network and Application Design Guide](#) for details about including scope information on configured host names.

Using remote execution clients in a z/OS UNIX environment

z/OS UNIX Remote Execution Protocol (z/OS UNIX REXEC) is a remote execution client that you can use to execute a command on a remote host and receive the results on the local host. z/OS UNIX RSH is also available as a remote execution client.

You must have the z/OS UNIX REXEC daemon (orexecd) running on the remote host to use z/OS UNIX REXEC to run z/OS UNIX commands. The z/OS UNIX REXEC client passes the user name, password, and command to the z/OS UNIX REXEC daemon. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

You do not have to have the z/OS UNIX REXEC daemon (orexecd) running on the remote host if you are not running z/OS UNIX commands. The z/OS UNIX REXEC client can use any REXEC daemon (not just z/OS UNIX).

You must have the z/OS UNIX RSH daemon (orshd) running on the remote host to use z/OS UNIX RSH to run z/OS UNIX commands. The z/OS UNIX RSH client passes the user name, password, and command to the z/OS UNIX RSH daemon. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

You do not have to have the z/OS UNIX RSH daemon (orshd) running on the remote host if you are not running z/OS UNIX commands. The z/OS UNIX RSH client can use any RSH daemon (not just z/OS UNIX).

The z/OS UNIX orexec/rexec command—Execute a command on the remote host

Purpose

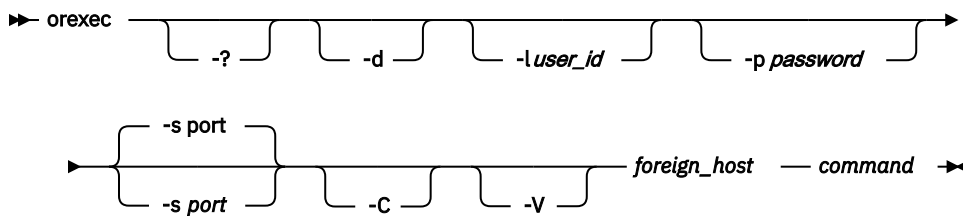
Use the z/OS UNIX `orexec/rexec` command to execute a command on the remote host and receive the results on the local host.

Notes:

1. The `rexec` command is a synonym for the `orexec` command in the z/OS UNIX shell. The `rexec` command syntax is the same as that for the `orexec` command.

2. If the `-s` parameter is not used to specify the port, the port to be used by the client must be defined in the `/etc/services` file as an `exec` entry defined to TCP. For information on `/etc/services`, Protocol Number and Port Assignments, see the [z/OS Communications Server: IP Configuration Reference](#).

Format



Note: Enter the `orexec` parameters `-d`, `-l`, `-p`, and `-s` in lowercase letters because they are case sensitive. The `user_id` and `password` parameters can be case sensitive, depending on the operating system of the remote host.

Parameters

-?

Displays the help message.

-d

Activates debug tracing.

-l *user_id*

Specifies the user ID on the foreign host.

-p *password*

Specifies the password for the user ID on the foreign host.

-s *port*

Specifies the TCP port number of the `rexec` server on the foreign host. The default is the port number defined in `/etc/services`.

-C

Forces messages to be displayed in uppercase characters.

-V

Displays the z/OS Communications Server version and release.

foreign_host

Specifies the name or IP address of the foreign host to which you are sending the `orexec` command. Specify the foreign host by its host name or IP address. When using IPv6 link-local addresses, scope information can be provided along with the name or IP address, as described in the `s` support for scope information in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

command

Specifies the command that is sent to the foreign host. The command is composed of one or more words. Coding is assigned after checking the prefixed parameters (`-l`, `-p`, `-s`) and assigning the remaining string as the command. The command you specify must not require a response from you to complete. The `orexec` command cannot interact with you after you enter data in the command format.

Examples

Use the `orexec` command to execute a command on a remote host:

```
orexec -l user28 -s 512 mvs1 lista
```

```
SYS1.HELP  
GIM.SGIMCLS0  
DSN230.DSNCLIST  
USER.CLIST  
BUILD.CLIST  
SYS1.HRFCLST  
USER28.ORSHD5.JOB00160.D0000103.?  
.
```

The z/OS UNIX orsh/rsh Command—Execute a Command on the remote host

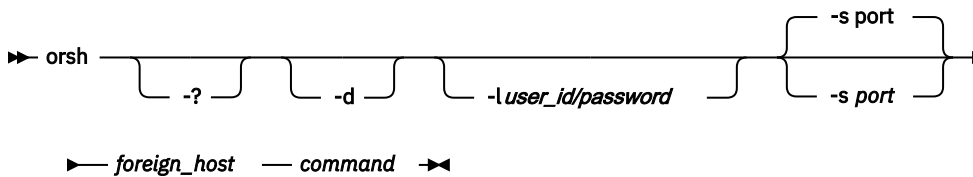
Purpose

Use the z/OS UNIX `orsh`/`rsh` command to execute a command on the remote host and receive the results on the local host.

Notes:

1. The `rsh` command is a synonym for the `orsh` command in the z/OS UNIX shell. The `rsh` command syntax is the same as the `orsh` command syntax.
2. If the `-s` parameter is not used to specify the port, the port to be used by the client must be defined in the `/etc/services` file as a shell entry defined to TCP. For information about `/etc/services`, Protocol Number, and Port Assignments, see the [z/OS Communications Server: IP Configuration Reference](#).

Format



Note: Enter the `orsh` parameters `-d`, `-l`, and `-s` in lowercase letters because they are case sensitive. The `user_id/password` parameter can be case sensitive, depending on the operating system of the remote host.

Parameters

-?

Displays the help message.

-d

Activates debug tracing.

-l user_id/password

Specifies the user ID and password. You must enter the slash (/) between the user ID and password if the target system is MVS. The character you must use can vary, depending on the target host. For VM hosts, you should use an @ character.

-s port

Specifies the TCP port number of the rsh server on the foreign host. The default is the port number defined in `/etc/services`.

foreign_host

Specifies the name or IP address of the foreign host to which you are sending the `orsh` command. Specify the foreign host by its host name or IP address. When using IPv6 link-local addresses, scope information can be provided along with the name or IP address, as described in the [support for scope information](#) in the [z/OS Communications Server: IPv6 Network and Application Design Guide](#).

command

Specifies the command that is sent to the foreign host. The command is composed of one or more words. Coding is assigned after checking the prefixed parameters (-l or -s) and assigning the remaining string as the command. The command you specify must not require a response from you to complete. The orsh command cannot interact with you after you enter data in the command format.

Examples

Use the orsh command to execute a command on a remote host:

```
orsh -l user28/password -s 512 mvs1 lista
```

```
SYS1.HELP  
GIM.SGIMCLS0  
DSN230.DSNCLIST  
USER.CLIST  
BUILD.CLIST  
SYS1.HRFCLST  
USER28.ORSHD5.JOB00160.D0000103.?  
.
```

Usage

Enter the required parameters on the command line. The orsh command does not prompt you for missing parameters.

Appendix A. Specifying data sets and files

This topic describes the file-naming formats for the following operating systems:

- “MVS data set and file naming” on page 359
- “AIX and UNIX file specifications” on page 362
- “AS/400 operating system file specifications” on page 363
- “VM file specifications” on page 363

Examples of each format are provided to show how the files appear to a TCP/IP user who is logged on to the different operating systems.

MVS data set and file naming

FTP subcommands can require a data set or file name. The format used to name a data set depends on the host system. Some systems limit the length of a data set name, and some systems are case sensitive.

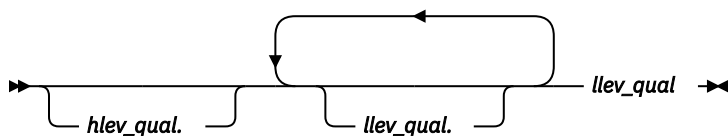
For information about the //DD file name syntax for FTP, see “ddname support with FTP” on page 56.

Data set names in MVS consist of one or more names, called qualifiers, each from one to eight characters long, that are delimited from one another by periods.

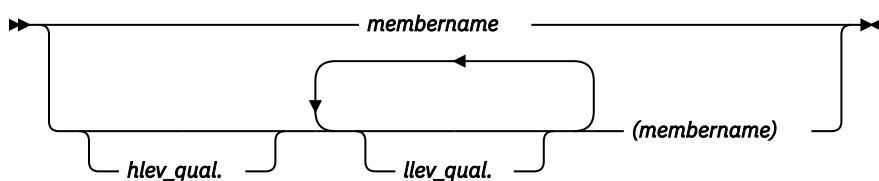
The leftmost qualifier in the data set name is the *high-level qualifier*. The rightmost qualifier in the data set name is the *low-level qualifier*. Partitioned data sets can be further qualified with a member name in the rightmost position. Qualifiers lying between them are called intermediate-level qualifiers.

For example, in the data set name `dog.bulldog.winston`, `dog` is the high-level qualifier, `bulldog` is the intermediate-level qualifier, and `winston` is the low-level qualifier.

Specify TSO sequential data sets in the following format:



Specify TSO partitioned data sets in the following format:



hlev_qual

Specifies the high-level qualifier of the data set. The default is the current working directory. If you specify this parameter, the complete data set name must be enclosed within single quotation marks (').

llev_qual

Specifies the low-level qualifier of the data set. You must specify this qualifier for sequential data sets.

membername

Specifies the member name of a partitioned data set (PDS). You must include parentheses around *membername* only when you also specify *llev_qual* or *hlev_qual*.

Sequential data set file naming

A sequential data set is a single file that can be allocated with any record length specified. The naming requirements for a sequential data set on an MVS host are minimal, and most of the requirements apply to any data set name under MVS.

The naming requirements for a sequential data set are:

- No part of the name can start with a numeric.
- No part of the name can be more than 8 characters in length.
- Each part of the name is separated by a period.
- A sequential data set name can have a minimum of 2 and a maximum of 44 characters.
- If single quotation marks (') are not used when specifying the data set name, the MVS system appends the current working directory as the first part of the name.

The following examples show the naming conventions for sequential data sets on an MVS host.

To access the sequential data set `KC00852.SEQ.NAMES`, the user, with the current working directory `KC00852`, enters one of the following:

- `'KC00852.SEQ.NAMES'`
- `SEQ.NAMES`

Either of these formats is acceptable for accessing a sequential data set.

Partitioned data set file naming

A partitioned data set (PDS) is a group of files contained in a library. The individual files that make up a PDS are called members. You can access an entire PDS or any individual member of a PDS.

The naming requirements for a partitioned data set are:

- No part of the name can start with a numeric.
- No part of the name can be more than eight characters in length.
- Each part of the name is separated by a period.
- If single quotation marks are not used when specifying the PDS name, the MVS system appends the current working directory as the first part of the name.

The difference between a sequential and partitioned data set specification is that the partitioned data set user accesses the directory of members in the PDS, and the sequential data set user accesses an individual file.

The following examples show the naming conventions for partitioned data sets on an MVS host.

To access the partitioned data set `KC00852.PDS.NAMES`, the user, with the current working directory `'KC00852'`, enters one of the following:

- `'KC00852.PDS.NAMES'`
- `PDS.NAMES`

Either of these formats is acceptable to access a partitioned data set.

Note: You can use the special character asterisk (*) as a global name character (wildcard) for pattern matching when you specify a data set name, with the following restrictions:

- The asterisk must be the last character, or the only character specified for a level of qualifier.
- When the data set name is enclosed in quotation marks, you cannot use the asterisk as a wildcard in the high-level qualifier of the data set name. Data set names not enclosed in quotation marks will use the setting of the current directory as the high-level qualifier.
- You can use the asterisk more than once in the complete data set name, but the asterisk must be the last character for each level of data set name qualifier.

- If you specify a member name, you cannot use an asterisk anywhere in the data set name.
- If you use an asterisk as all or part of the member name, you cannot use an asterisk anywhere else in the data set name.

To access an individual member of a PDS, the member name is entered in parentheses.

To access the member PROPER in the PDS KC00852.PDS.NAMES, the user, with the current working directory KC00852, enters one of the following:

- 'KC00852.PDS.NAMES(PROPER)'
- PDS.NAMES(PROPER)

Either of these formats is acceptable to access an individual member of a partitioned data set.

Transferring data between partitioned and sequential data sets

When transferring data between partitioned and sequential data sets, ensure that the *local_file* and *foreign_file* parameters of the FTP subcommands are compatible with the type of data set you are transferring to or from. For example, if your local working directory is a partitioned data set and you want to GET the sequential file TEST.FILE1, you cannot issue the subcommand GET TEST.FILE1 to retrieve the file, because this subcommand by default tries to use the local PDS member name TEST.FILE1, which is not a valid member name.

To keep the local and remote file names compatible with the type of data set used, do one of the following:

- Specify both the local and remote file names

For example:

```
GET TEST.FILE TESTFL1
PUT TESTPDS(FILE1) FILE1
```

- Change the directory to the lowest level qualifier.

For example, to transfer between the PDS 'USER14.TESTPDS(NAME1)' and the sequential data set 'USER17.SEQ.NAME1', do the following:

1. Change the local directory (LCD) to 'USER14.TESTPDS'
2. Change the directory (CD) to 'USER17.SEQ'
3. Enter one of the following:
 - GET NAME1 to get 'USER17.SEQ.NAME1' as 'USER14.TESTPDS(NAME1)'
 - PUT NAME1 to put 'USER14.TESTPDS(NAME1)' as 'USER17.SEQ.NAME1'

Data transfer methods

You must use the appropriate transmission attributes to preserve the content and structure of the data when you transfer data sets or files between two hosts. Use the FTP MODE subcommand to specify how the bits of data are to be transmitted, and the FTP TYPE subcommand to define the way that data is represented during the data transfer.

See “MOde subcommand—Set the data transfer mode” on page 220 for information about the MODE subcommand, and “TYpe subcommand—Set the data transfer type” on page 289 for information about the TYPE subcommand.

TCP/IP supports only the data transfer of a data set or file structured as a continuous sequence of data bytes. This ensures that the correct record format is preserved across MVS hosts.

Table 27 on page 362 shows how to set the transmission attributes for different host systems. IBM mainframe operating systems (VM or MVS) are identified as EBCDIC transfer types. Systems with ASCII storage are identified as ASCII transfer types. A text file of an ASCII transfer type contains standard, displayable characters; a carriage return (ASCII X'0D' and EBCDIC X'15'); and line feed characters (ASCII

X'0A' and EBCDIC X'25'). A text file of an EBCDIC transfer type contains standard, displayable characters only. A binary file can contain any characters.

Table 27. Recommended methods for data transfer

| Transfer between host types | Transfer type | Mode |
|-----------------------------------|----------------|--------|
| EBCDIC to EBCDIC—text data | EBCDIC | Stream |
| EBCDIC to EBCDIC—binary data | EBCDIC | Block |
| EBCDIC to ASCII—text data | ASCII | Stream |
| ASCII to EBCDIC—text data | ASCII | Stream |
| ASCII to EBCDIC—binary data | Image (binary) | Stream |
| ASCII to EBCDIC to ASCII—all data | Image (binary) | Stream |

Note: For the data transfer of "ASCII to EBCDIC to ASCII—all data", the EBCDIC host is used for storage only. Data is not used on the EBCDIC host.

Transferring PDS directory information

When a PDS member is transmitted in block or compressed data transfer mode with a representation type of EBCDIC, the user data associated with the PDS member is also transferred to the directory on the target host. This transfer occurs only when using an MVS client. No PDS directory information is transferred if the member is null (empty).

AIX and UNIX file specifications

For the Advanced Interactive Executive (AIX) and UNIX operating systems, data is stored in files. Related files are stored in a directory. z/OS Communications Server files are UNIX files.

Specify AIX and UNIX files in the following format:

➤ / — *directory* — / — *filename* ➤

directory

Specifies a directory name. Directories contain the names of files, other directories, or both.

filename

Specifies a file name. It can be up to 14 characters long.

The complete name of an AIX and UNIX file contains the directory name and the file name. See the following example:

```
/mailfiles/cooks
```

Where:

mailfiles

The directory name

cooks

The file name

In the AIX and UNIX operating systems, you specify the first slash (/) only when you begin at the root directory. If you are specifying a file in the current directory, enter only the file name. For example, if you are in the current directory *mailfiles* and you want to access the *cooks* file, specify:

```
cooks
```

The directory name and file name can each be up to 14 characters in length. The AIX and UNIX operating systems distinguish between uppercase and lowercase letters in file names.

Requirement: A directory name and file name should not include characters that have a special meaning to the shell, such as backslash (\), ampersand (&), and period (.).

Tips:

- If a z/OS UNIX file name contains any single quotation marks ('), enclose the name in double quotation marks (").
- If a z/OS UNIX file name contains any double quotation marks ("), enclose the name in single quotation marks (').
- If a z/OS UNIX file name does not contain any quotation marks, do not enclose the name in quotation marks.

AS/400 operating system file specifications

For the AS/400 operating system, data is stored in files.

Specify AS/400 files in the following format:

➤ *library* — / — *file* — . — *member* ➤

library

A library name. Libraries contain the names of programs, files, and commands.

file.member

The file name.

In the AS/400 operating system, files can have one or more members. Each file can consist of data records, source programs, or database definitions.

The FTP subcommand PUT is used to copy a local file member into a file at the remote host. See the following example:

```
PUT PDS.DATA(MBR1) LIB1/FILEA.MBR1
```

In this example, the PUT subcommand copies member MBR1 from the partitioned data set PDS.DATA on the local host to member MBR1 of file FILEA in the library LIB1 on the remote host. If the member already exists at the remote host, it is overwritten.

VM file specifications

Data is stored in files on VM hosts. Specify VM files in the following format:

➤ *filename* — . — *filetype* ➤

filename

Specifies the file name.

filetype

Specifies the file type.

Note: The file mode is not accepted by foreign VM hosts; it is taken to be the file mode associated with the current working directory. The file mode is not used in TCP/IP commands.

For example, if you want to specify a file named accounts with a file type cprog, enter the following:

```
accounts.cprog
```

Where *filename* is accounts and *filetype* is cprog.

All VM file specifications are treated as if they are entered in uppercase. The file name and the file type consist of 1 to 8 alphanumeric characters. Other valid characters are \$, #, @ (at character), + (plus), - (hyphen), and _ (underscore).

You can use the special character * (asterisk) for pattern matching.

Appendix B. Mapping values for the APL2 character set

This topic lists the GDDMXD/MVS default mapping values for the APL2 character set. However, If the *hlq.GDXAPLCS.MAP* data set exists, the default mapping values are overridden.

Each entry in the *hlq.GDXAPLCS.MAP* data set (alternative character set) contains the mapping for a particular physical key that corresponds to three characters. The characters correspond to the physical key by:

- Pressing the key alone
- Pressing the key and the **Shift** key simultaneously
- Pressing the key and the **Alt** key simultaneously

The *hlq.GDXAPLCS.MAP* data set entries must contain the following seven single-byte hexadecimal values entered as EBCDIC characters:

- Value 1 is the hexadecimal keycode for the physical key.
- Values 2, 4, and 6 identify whether the character is in the primary or alternative character set for the emulated 3179G. If the character is in the primary set, the value is 0; if the character is in the alternative set, the value is 8.
- Values 3, 5, and 7 specify the EBCDIC code of the character in the character set.

The combination of values 2 and 3 define the bytes that describe the character when the key corresponding to the keycode is pressed alone.

The combination of values 4 and 5 define the bytes that describe the character when the key corresponding to the keycode and the **Shift** key are pressed simultaneously.

The combination of values 6 and 7 define the bytes that describe the character when the key corresponding to the keycode and the **Alt** key are pressed simultaneously.

Table 28 on page 365 lists the mapping values for the APL2 character set.

Table 28. Mapping values for the APL2 character set

| Character name | Character set value | EBCDIC value | Default keycode |
|-------------------|---------------------|--------------|------------------|
| Quad Jot | 8 | 73 | 9 + Shift |
| Quad Slope | 8 | CE | 9 + Alt |
| 1 | 0 | F1 | A |
| Diaeresis | 8 | 72 | A + Shift |
| Down Tack Up Tack | 8 | DA | A + Alt |
| 2 | 0 | F2 | B |
| Overbar | 8 | A0 | B + Shift |
| Del Tilde | 8 | FB | B + Alt |
| 3 | 0 | F3 | C |
| < | 0 | 4C | C + Shift |
| Del Stile | 8 | DC | C + Alt |
| 4 | 0 | F4 | D |

Table 28. Mapping values for the APL2 character set (continued)

| Character name | Character set value | EBCDIC value | Default keycode |
|------------------|---------------------|--------------|-------------------|
| Not Greater | 8 | 8C | D + Shift |
| Delta Stile | 8 | DD | D + Alt |
| 5 | 0 | F5 | E |
| = | 0 | 7E | E + Shift |
| Circle Stile | 8 | CD | E + Alt |
| 6 | 0 | F6 | F |
| Not Less | 8 | AE | F + Shift |
| Circle Slope | 8 | CF | F + Alt |
| 7 | 0 | F7 | 10 |
| > | 0 | 6E | 10 + Shift |
| Circle Bar | 8 | ED | 10 + Alt |
| 8 | 0 | F8 | 11 |
| Not Equal | 8 | BE | 11 + Shift |
| Circle Star | 8 | FD | 11 + Alt |
| 9 | 0 | F9 | 12 |
| Down Caret | 8 | 78 | 12 + Shift |
| Down Caret Tilde | 8 | CB | 12 + Alt |
| 0 | 0 | F0 | 13 |
| Up Caret | 8 | 71 | 13 + Shift |
| Up Caret Tilde | 8 | CA | 13 + Alt |
| + | 0 | 4E | 14 |
| - | 0 | 60 | 14 + Shift |
| ! | 8 | DB | 14 + Alt |
| Times | 8 | B6 | 15 |
| Divide | 8 | B8 | 15 + Shift |
| Quad Divide | 8 | EE | 15 + Alt |
| Q | 0 | D8 | 19 |
| ? | 0 | 6F | 19 + Shift |
| Q Underbar | 8 | 58 | 19 + Alt |
| W | 0 | E6 | 1A |
| Omega | 8 | B4 | 1A + Shift |
| W Underbar | 8 | 66 | 1A + Alt |
| E | 0 | C5 | 1B |
| Epsilon | 8 | B1 | 1B + Shift |

Table 28. Mapping values for the APL2 character set (continued)

| Character name | Character set value | EBCDIC value | Default keycode |
|--------------------|---------------------|--------------|-------------------|
| E Underbar | 8 | 45 | 1B + Alt |
| R | 0 | D9 | 1C |
| Rho | 8 | B3 | 1C + Shift |
| R Underbar | 8 | 59 | 1C + Alt |
| T | 0 | E3 | 1D |
| Tilde | 8 | 80 | 1D + Shift |
| T Underbar | 8 | 63 | 1D + Alt |
| Y | 0 | E8 | 1E |
| Up Arrow | 8 | 8A | 1E + Shift |
| Y Underbar | 8 | 68 | 1E + Alt |
| U | 0 | E4 | 1F |
| Down Arrow | 8 | 8B | 1F + Shift |
| U Underbar | 8 | 64 | 1F + Alt |
| I | 0 | C9 | 20 |
| Iota | 8 | B2 | 20 + Shift |
| I Underbar | 8 | 49 | 20 + Alt |
| O | 0 | D6 | 21 |
| Circle | 8 | 9D | 21 + Shift |
| O Underbar | 8 | 56 | 21 + Alt |
| P | 0 | D7 | 22 |
| Star | 0 | 5C | 22 + Shift |
| P Underbar | 8 | 57 | 22 + Alt |
| Left Arrow | 8 | 9F | 23 |
| Right Arrow | 8 | 8F | 23 + Shift |
| Quad Quote | 8 | DE | 23 + Alt |
| Left Brk Right Brk | 8 | CC | 24 |
| Iota Underbar | 8 | 74 | 24 + Shift |
| Delta Underbar | 8 | FC | 24 + Alt |
| Equal Underbar | 8 | E1 | 25 |
| Epsilon Underbar | 8 | E1 | 25 + Shift |
| Diaeresis Dot | 8 | 75 | 25 + Alt |
| A | 0 | C1 | 27 |
| Alpha | 8 | B0 | 27 + Shift |
| A Underbar | 8 | 41 | 27 + Alt |

Table 28. Mapping values for the APL2 character set (continued)

| Character name | Character set value | EBCDIC value | Default keycode |
|----------------|---------------------|--------------|-------------------|
| S | 0 | E2 | 28 |
| Up Stile | 8 | 8D | 28 + Shift |
| S Underbar | 8 | 62 | 28 + Alt |
| D | 0 | C4 | 29 |
| Down Stile | 8 | 8E | 29 + Shift |
| D Underbar | 8 | 44 | 29 + Alt |
| F | 0 | C6 | 2A |
| Underbar | 0 | 6D | 2A + Shift |
| F Underbar | 8 | 46 | 2A + Alt |
| G | 0 | C7 | 2B |
| Del | 8 | BA | 2B + Shift |
| G Underbar | 8 | 47 | 2B + Alt |
| H | 0 | C8 | 2C |
| Delta | 8 | BB | 2C + Shift |
| H Underbar | 8 | 48 | 2C + Alt |
| J | 0 | D1 | 2D |
| Jot | 8 | AF | 2D + Shift |
| J Underbar | 8 | 51 | 2D + Alt |
| K | 0 | D2 | 2E |
| Quote | 0 | 7D | 2E + Shift |
| K Underbar | 8 | 52 | 2E + Alt |
| L | 0 | D3 | 2F |
| Quad | 8 | 90 | 2F + Shift |
| L Underbar | 8 | 53 | 2F + Alt |
| Left Bracket | 8 | AD | 30 |
| (| 0 | 4D | 30 + Shift |
| Down Tack Jot | 8 | FE | 30 + Alt |
| Right Bracket | 8 | BD | 31 |
|) | 0 | 5D | 31 + Shift |
| Up Tack Jot | 8 | EF | 31 + Alt |
| Z | 0 | E9 | 36 |
| Left Shoe | 8 | 9B | 36 + Shift |
| Z Underbar | 8 | 69 | 36 + Alt |
| X | 0 | E7 | 37 |

Table 28. Mapping values for the APL2 character set (continued)

| Character name | Character set value | EBCDIC value | Default keycode |
|----------------|---------------------|--------------|-------------------|
| Right Shoe | 8 | 9A | 37 + Shift |
| X Underbar | 8 | 67 | 37 + Alt |
| C | 0 | C3 | 38 |
| Up Shoe | 8 | AA | 38 + Shift |
| C Underbar | 8 | 43 | 38 + Alt |
| V | 0 | E5 | 39 |
| Down Shoe | 8 | AB | 39 + Shift |
| V Underbar | 8 | 65 | 39 + Alt |
| B | 0 | C2 | 3A |
| Down Tack | 8 | AC | 3A + Shift |
| B Underbar | 8 | 42 | 3A + Alt |
| N | 0 | D5 | 3B |
| Up Tack | 8 | BC | 3B + Shift |
| N Underbar | 8 | 55 | 3B + Alt |
| M | 0 | D4 | 3C |
| Stile | 0 | 4F | 3C + Shift |
| M Underbar | 8 | 54 | 3C + Alt |
| , | 0 | 6B | 3D |
| ; | 0 | 5E | 3D + Shift |
| Up Shoe Jot | 8 | DF | 3D + Alt |
| period | 0 | 4B | 3E |
| : | 0 | 7A | 3E + Shift |
| Slope Bar | 8 | EB | 3E + Alt |
| / | 0 | 61 | 3F |
| \ | 0 | E0 | 3F + Shift |
| Slash Bar | 8 | EA | 3F + Alt |
| Space | 0 | 40 | 45 |

Appendix C. TELNET extensions

This topic describes the Telnet 3270 DBCS Transform special operations. The following sections are included:

- [“Character set cross reference table” on page 371](#)
- [“Special key operation for TELNET” on page 373](#)
- [“Operation of PF and PA keys with TELNET” on page 375](#)
- [“Sense codes for special key operation with TELNET” on page 377](#)

Character set cross reference table

Table 29 on page 371 describes the language, codefiles, and character sets for the Telnet 3270 DBCS Transform extended language support.

| Table 29. TCP/IP character set cross reference | | | | | | |
|--|----------------------|---------------------------------------|---------------|-----------------------------|--------------------|--------------------|
| Keyword | Codefiles | Description | Character set | Description | CCSID ¹ | CPGID ² |
| KANJI | | | | | | |
| JIS78KJ | J8EETA J8EATE | JIS 8 Bit English SBCS | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | English SBCS | none | none |
| | J8KETA J8KATE | JIS 8 Bit Katakana SBCS | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | Katakana SBCS | none | none |
| | JIS78ETA JIS78ATE | JIS 1978 Kanji DBCS | ASCII | JIS X0208 1978 | 00955 | 00955 |
| | | | EBCDIC | Japanese Host DBCS | 00300 | 00300 |
| JIS83KJ | J8EETA J8EATE | JIS 8 Bit English SBCSI | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | English SBCS | none | none |
| | J8KETA J8KATE | JIS 8 Bit Katakana SBCS | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | Katakana SBCS | none | none |
| | JIS83ETA JIS83ATE | JIS 1983 Kanji DBCS | ASCII | JIS X0208 1990 | 00952 | 00952 |
| | | | EBCDIC | Japanese Host DBCS | 00300 | 00300 |
| SJISKANJ I | A8EETA A8EATE | 8 Bit English SBCS ASCII | ASCII | ISO/ANSI Multilingual | 00819 | 00819 |
| | | | EBCDIC | Japanese Latin Host SBCS | 01027 | 01027 |
| | A8KETA A8KATE | 8 Bit Katakana SBCS | ASCII | ISO/ANSI Multilingual | 00819 | 00819 |
| | | | EBCDIC | Japanese Katakana Host SBCS | 00290 | 00290 |
| | SJISETA SJISATE | SJIS 1978 Kanji DBCS from EZAKJLAT | ASCII | SJIS 0941 | 0300 | 0300 |
| | | | EBCDIC | | 0941 | 0941 |
| | SJISETA SJISATE | SJIS 1995 Kanji DBCS from EZAKJ941 | ASCII | SJIS 0941 | 0300 | 0300 |
| | | | EBCDIC | | 0941 | 0941 |

Table 29. TCP/IP character set cross reference (continued)

| Keyword | Codefiles | Description | Character set | Description | CCSID ¹ | CPGID ² |
|----------|----------------------|------------------------------------|---------------|-----------------------------|--------------------|--------------------|
| DECKANJI | SJDCEETA SJDCEATE | DEC English SBCS | ASCII | | none | none |
| | | | EBCDIC | | none | none |
| | SJDCKETA SJDCKATE | DEC Katakana SBCS | ASCII | | none | none |
| | | | EBCDIC | | none | none |
| | JDECETA JDECATE | DEC Kanji DBCS | ASCII | | none | none |
| | | | EBCDIC | Japanese Host DBCS | none | none |
| EUCKANJI | SJECEETA SJECEATE | 8 Bit English SBCS | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | Japanese Latin Host SBCS | 01027 | 01027 |
| | SJECKETA SJECKATE | 8 Bit Katakana SBCS | ASCII | JIS X0201 8 Bit | none | none |
| | | | EBCDIC | Japanese Katakana Host SBCS | 00290 | 00290 |
| | JEUCETA JEUCATE | Japanese Extended Unix DBCS | ASCII | JIS X0208 1990 | 00952 | 00952 |
| | | | EBCDIC | Japanese Host DBCS | 00300 | 00300 |
| HANGEUL | | | | | | |
| KSC5601 | SKSHETA SKSHATE | Korean Standard Code KSC 5601 SBCS | ASCII | KSC 5601 SBCS | 01088 | 01088 |
| | | | EBCDIC | Korean Host SBCS | 00833 | 00833 |
| | KSHETA KSHATE | Korean Standard Code KSC 5601 DBCS | ASCII | KSC 5601 DBCS | 00951 | 00951 |
| | | | EBCDIC | Korean Host DBCS | 00834 | 00834 |
| HANGEUL | SHANETA SHANATE | Hangeul SBCS | ASCII | Korean PC SBCS | 00891 | 00891 |
| | | | EBCDIC | Korean Host SBCS | 00833 | 00833 |
| | HANETA HANATE | Hangeul DBCS | ASCII | Korean PC DBCS | 00926 | 00926 |
| | | | EBCDIC | Korean Host DBCS | 00834 | 00834 |
| TCHINESE | | | | | | |
| TCHINESE | STCHETA STCHATE | Traditional Chinese SBCS | ASCII | T-Chinese PC SBCS | 00904 | 00904 |
| | | | EBCDIC | CECP Host SBCS | 00037 | 00037 |
| | TCHETA TCHATE | Traditional Chinese DBCS | ASCII I | T-Chinese PC DBCS | 00927 | 00927 |
| | | | EBCDIC | T-Chinese Host DBCS | 00835 | 00835 |
| BIG5 | SBG5ETA SBG5ATE | Big-5 Chinese SBCS I | ASCII I | Big-5 Chinese PC SBCS | 01114 | 01114 |
| | | | EBCDIC | CECP Host SBCS | 00037 | 00037 |
| | BG5ETA BG5ATE | Big-5 Chinese DBCS | ASCII | Big-5 PC DBCS | 00947 | 00947 |
| | | | EBCDIC | T-Chinese Host DBCS | 00835 | 00835 |
| SCHINESE | | | | | | |

| Table 29. TCP/IP character set cross reference (continued) | | | | | | |
|--|--------------------|-------------------------|---------------|---------------------|--------------------|--------------------|
| Keyword | Codefiles | Description | Character set | Description | CCSID ¹ | CPGID ² |
| SCHINESE | SSCHETA SSCHATE | Simplified Chinese SBCS | ASCII | S-Chinese PC SBCS | 01115 | 01115 |
| | | | EBCDIC | S-Chinese Host SBCS | 00836 | 00836 |
| | SCHETA SCHATE | Simplified Chinese DBCS | ASCII | S-Chinese PC DBCS | 01380 | 01380 |
| | | | EBCDIC | S-Chinese Host DBCS | 00837 | 00837 |
| ¹ Coded Character Set ID | | | | | | |
| ² Code Page Group ID | | | | | | |

Special key operation for TELNET

Table 30 on page 373 contains information about the operation of special keys for the TELNET function when you are using a terminal that is not part of the 3270 family.

Use these key combinations if you are using TELNET and your terminal does not have the key that you want to use. For example, if you want to clear your screen and your terminal does not have an ERASE INPUT key, press CTRL+Y to get the same result.

To use the TELNET extensions, TCP/IP must be configured using the DBCSTRANSFORM option. For more information, see the [z/OS Communications Server: IP Configuration Reference](#).

Table 30. Special key conversions

| Function name | Input keys | Function description |
|---------------|------------|--|
| Duplicate | Ctrl+D | Press CTRL and D together. This combination enters the Duplicate control code in the screen buffer and a TAB is performed on the screen. |
| Field Mark | Ctrl+K | Press CTRL and K together. This combination enters the Field Mark control code in the screen buffer and displays it as a blank on the screen. |
| Redisplay | Ctrl+V | Press CTRL and V together. This combination redisplays the contents of the screen buffer on your screen. |
| Erase Input | Ctrl+Y | Press CTRL and Y together. This combination erases all characters in the unprotected fields on the screen and replaces them with blanks. The cursor is placed at the first unprotected character position on the screen. |
| Erase EOF | Ctrl+X | Press CTRL and X together. This combination erases all characters in an unprotected field from the cursor position to the end of the field and replaces them with blanks. If the cursor is on a protected field, the screen is inhibited and no characters are erased. |

Table 30. Special key conversions (continued)

| Function name | Input keys | Function description |
|-----------------------------------|----------------------------|--|
| Delete One Character | Del | The DEL deletes the character at the cursor position, if the field is unprotected. The cursor does not move. All characters in the unprotected field to the right of the cursor are shifted one position to the left and blank characters are added at the end of the field. |
| Alphanumeric or Alphanumeric-Kana | Ctrl+B | Press CTRL and B together. This combination is a toggle switch that redisplayes the screen by switching between Alphanumeric and Alphanumeric-Kana mode. |
| Field Forward Tab | Ctrl+F | Press CTRL and F together. This combination moves the cursor to the first character position in the next unprotected field. If the screen is unformatted or there are no unprotected fields on the screen, the cursor is placed in the first position on the screen. |
| Field Backward Tab | Ctrl+A F, or Ctrl+A Ctrl+F | Press CTRL and A together then press F, or Press CTRL and A together then press CTRL and F together. This combination moves the cursor to the first character position in the previous unprotected field. If the screen is unformatted or there are no unprotected fields on the screen, the cursor is placed in the first position on the screen. |
| Home | CSI P, or ESC [P | Press CSI then press P, or Press ESC then press [then press P. This combination moves the cursor to the first character position in the first unprotected field on the screen. If the screen is unformatted or there are no unprotected fields on the screen, the cursor is placed in the first character position on the screen. |
| Move Cursor Up | CSI A, or ESC [A | Press CSI then press A, or press ESC then press [then press A. This combination moves the cursor up one line in the same column. If the cursor is on the first line on the screen, it moves to the last line on the screen. |
| Move Cursor Down | CSI B, or ESC [B | Press CSI then press B, or press ESC then press [then press B. This combination moves the cursor down one line in the same column. If the cursor is on the last line on the screen, it moves to the first line on the screen. |
| Move Cursor Right | CSI C, or ESC [C | Press CSI then press C, or press ESC then press [then press C. This combination moves the cursor one character to the right. If the cursor is in the last column in a line, it moves to the first position in the next line on the screen. If the cursor is in the last position on the screen, it moves to the first position on the screen. |

Table 30. Special key conversions (continued)

| Function name | Input keys | Function description |
|-------------------------|----------------------------|---|
| Move Cursor Left | CSI D, or ESC [D | Press CSI then press D, or press ESC then press [then press D. This combination moves the cursor one character to the left. If the cursor is in the first column in a line, it moves to the last position in the previous line on the screen. If the cursor is in the first position on the screen, it moves to the last position on the screen. |
| Backspace One Character | Ctrl+H | Press CTRL and H together. This combination deletes one character before the cursor position in an unprotected field. The cursor moves one position to the left and all characters in the field shift one position to the left. If the cursor is on a protected field the screen is inhibited. |
| Reset | Ctrl+R | Press CTRL and R together. This combination releases the screen inhibit condition. When the screen is inhibited, only the RESET and MASTER RESET key combinations remain active. |
| Master Reset | Ctrl+A M, or Ctrl+A Ctrl+M | Press CTRL and A together then press M, or Press CTRL and A together then press CTRL and M together. This combination produces the same results that you get if you key in REDISPLAY followed by RESET. |
| Clear | Ctrl+L | Press CTRL and L together. This combination fills the screen with blanks and places the cursor at the first character position on the screen. MASTER RESET key combinations remain active. |
| Enter | Ctrl+M | Press CTRL and M together. This combination sends the data on the screen to the host system. |

CSI stands for Control Sequence Indicator.

Operation of PF and PA keys with TELNET

This section describes the PF and PA keys for TELNET operations when you are using a terminal that is not part of the 3270 family.

When you press a key combination, a code that represents the 3270 equivalent symbol is sent to the application with which you have established TELNET communication. The application that you are using controls how these codes are used.

PF Key

Input Keys

PF1

Press ESC then press 1

PF2

Press ESC then press 2

PF3

Press ESC then press 3

PF4

Press ESC then press 4

PF5

Press ESC then press 5

PF6

Press ESC then press 6

PF7

Press ESC then press 7

PF8

Press ESC then press 8

PF9

Press ESC then press 9

PF10

Press ESC then press 0

PF11

Press ESC then press the period key (.)

PF12

Press ESC then press =

PF13

Press ESC then press 1

PF14

Press ESC then press 2

PF15

Press ESC then press 3

PF16

Press ESC then press 4

PF17

Press ESC then press 5

PF18

Press ESC then press 6

PF19

Press ESC then press 7

PF20

Press ESC then press 8

PF21

Press ESC then press 9

PF22

Press ESC then press 0

PF23

Press ESC then press .

PF24

Press ESC then press =

PA Key**Input Keys****PA1**

Press Ctrl+P then press 1

PA2

Press Ctrl+P then press 2

PA3

Press Ctrl+P then press 3

Sense codes for special key operation with TELNET

Table 31 on page 377 describes the sense codes that are returned with the error messages for special key operations when you are using TELNET.

For information about error messages, see *z/OS Communications Server: IP Messages Volume 1 (EZA)*.

Table 31. Sense codes

| Sense code | Problem description |
|------------|---|
| 8001 | The command data length is less than 0 bytes. |
| 8002 | There is not enough data in an ERASE/WRITE or an ERASE/WRITE ALTERNATE command. |
| 8003 | There is not enough data in A WRITE command. |
| 8004 | There is not enough data in A WRITE STRUCTURED FIELD command. |
| 21001 | WCC is not a character. The error occurred in a WRITE command. |
| 21002 | There is not enough data in a START FIELD subcommand. The error occurred in a WRITE command. |
| 21003 | There is not enough data in a START FIELD EXTENDED subcommand. The error occurred in a WRITE command. |
| 21004 | There is not enough data in a MODIFY FIELD subcommand. The error occurred in a WRITE command. |
| 21005 | There is not enough data in a SET BUFFER ADDRESS subcommand. The error occurred in a WRITE command. |
| 21006 | There is not enough data in a REPEAT TO ADDRESS subcommand. The error occurred in a WRITE command. |
| 21007 | There is not enough data in an ERASE UNPROTECTED TO ADDRESS subcommand. The error occurred in a WRITE command. |
| 22001 | The specified attribute is not a character. The error occurred in a START FIELD subcommand. |
| 22002 | A current buffer addressing error occurred. The error occurred in a START FIELD subcommand. |
| 23001 | An addressing error for the current buffer occurred. The error occurred in a START FIELD EXTENDED subcommand. |
| 23002 | The specified attribute is not a character. The error occurred in a START FIELD EXTENDED subcommand. |
| 23003 | The specified attribute is not an acceptable attribute type. The error occurred in a START FIELD EXTENDED subcommand. |
| 24001 | A current buffer addressing error occurred. The error occurred in a MODIFY FIELD subcommand. |
| 24002 | The specified attribute is not a character. The error occurred in a MODIFY FIELD subcommand. |

Table 31. Sense codes (continued)

| Sense code | Problem description |
|------------|---|
| 24003 | The specified attribute is not an acceptable attribute type. The error occurred in a MODIFY FIELD subcommand. |
| 25001 | The specified address is incorrect. The error occurred in a SET BUFFER ADDRESS subcommand. |
| 26001 | A current buffer addressing error occurred. The error occurred in a PROGRAM TAB subcommand. |
| 28001 | A current buffer addressing error occurred. The error occurred in a REPEAT TO ADDRESS subcommand. |
| 28002 | The specified address is incorrect. The error occurred in a REPEAT TO ADDRESS subcommand. |
| 29001 | A current buffer addressing error occurred. The error occurred in an ERASE UNPROTECTED TO ADDRESS subcommand. |
| 29002 | The specified address is incorrect. The error occurred in an ERASE UNPROTECTED TO ADDRESS subcommand. |
| 30001 | A current buffer addressing error occurred. The error occurred in the Write Data Process. |
| 36001 | Another structured field appeared after the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 36002 | The length is too long or too short in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 36003 | A reserved character is incorrect in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 36004 | The type is incorrect in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 38001 | The length is too long or too short in the ERASE/RESET field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 38002 | The Partition ID is incorrect in the ERASE/RESET field. The error occurred in a WRITE STRUCTURED FIELD command. |
| 39001 | The length is too long or too short in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command. |
| 39002 | The Partition ID is incorrect in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command. |
| 39003 | The Reply Mode is incorrect in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command. |
| 40001 | <p>The 3270 outbound data stream contains a command that is not in this list.</p> <ul style="list-style-type: none"> • WRITE • ERASE/WRITE • ERASE/WRITE ALTERNATE • ERASE ALL UNPROTECTED <p>The error occurred in a WRITE STRUCTURED FIELD command.</p> |

Table 31. Sense codes (continued)

| Sense code | Problem description |
|-------------------|--|
| 50001 | SO/SI is on longer a pair. SET REPLY mode. The error occurred in the GRFTOMAP Process. |

Appendix D. Related protocol specifications

This appendix lists the related protocol specifications (RFCs) for TCP/IP. The Internet Protocol suite is still evolving through requests for comments (RFC). New protocols are being designed and implemented by researchers and are brought to the attention of the Internet community in the form of RFCs. Some of these protocols are so useful that they become recommended protocols. That is, all future implementations for TCP/IP are recommended to implement these particular functions or protocols. These become the *de facto* standards, on which the TCP/IP protocol suite is built.

RFCs are available at <http://www.rfc-editor.org/rfc.html>.

Draft RFCs that have been implemented in this and previous Communications Server releases are listed at the end of this topic.

Many features of TCP/IP Services are based on the following RFCs:

RFC

Title and Author

RFC 652

Telnet output carriage-return disposition option D. Crocker

RFC 653

Telnet output horizontal tabstops option D. Crocker

RFC 654

Telnet output horizontal tab disposition option D. Crocker

RFC 655

Telnet output formfeed disposition option D. Crocker

RFC 657

Telnet output vertical tab disposition option D. Crocker

RFC 658

Telnet output linefeed disposition D. Crocker

RFC 698

Telnet extended ASCII option T. Mock

RFC 726

Remote Controlled Transmission and Echoing Telnet option J. Postel, D. Crocker

RFC 727

Telnet logout option M.R. Crispin

RFC 732

Telnet Data Entry Terminal option J.D. Day

RFC 733

Standard for the format of ARPA network text messages D. Crocker, J. Vittal, K.T. Pogran, D.A. Henderson

RFC 734

SUPDUP Protocol M.R. Crispin

RFC 735

Revised Telnet byte macro option D. Crocker, R.H. Gumpertz

RFC 736

Telnet SUPDUP option M.R. Crispin

RFC 749

Telnet SUPDUP—Output option B. Greenberg

RFC 765

File Transfer Protocol specification J. Postel

RFC 768

User Datagram Protocol J. Postel

RFC 779

Telnet send-location option E. Killian

RFC 791

Internet Protocol J. Postel

RFC 792

Internet Control Message Protocol J. Postel

RFC 793

Transmission Control Protocol J. Postel

RFC 820

Assigned numbers J. Postel

RFC 823

DARPA Internet gateway R. Hinden, A. Sheltzer

RFC 826

Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware D. Plummer

RFC 854

Telnet Protocol Specification J. Postel, J. Reynolds

RFC 855

Telnet Option Specification J. Postel, J. Reynolds

RFC 856

Telnet Binary Transmission J. Postel, J. Reynolds

RFC 857

Telnet Echo Option J. Postel, J. Reynolds

RFC 858

Telnet Suppress Go Ahead Option J. Postel, J. Reynolds

RFC 859

Telnet Status Option J. Postel, J. Reynolds

RFC 860

Telnet Timing Mark Option J. Postel, J. Reynolds

RFC 861

Telnet Extended Options: List Option J. Postel, J. Reynolds

RFC 862

Echo Protocol J. Postel

RFC 863

Discard Protocol J. Postel

RFC 864

Character Generator Protocol J. Postel

RFC 865

Quote of the Day Protocol J. Postel

RFC 868

Time Protocol J. Postel, K. Harrenstien

RFC 877

Standard for the transmission of IP datagrams over public data networks J.T. Korb

RFC 883

Domain names: Implementation specification P.V. Mockapetris

RFC 884

Telnet terminal type option M. Solomon, E. Wimmers

- RFC 885**
Telnet end of record option J. Postel
- RFC 894**
Standard for the transmission of IP datagrams over Ethernet networks C. Hornig
- RFC 896**
Congestion control in IP/TCP internetworks J. Nagle
- RFC 903**
Reverse Address Resolution Protocol R. Finlayson, T. Mann, J. Mogul, M. Theimer
- RFC 904**
Exterior Gateway Protocol formal specification D. Mills
- RFC 919**
Broadcasting Internet Datagrams J. Mogul
- RFC 922**
Broadcasting Internet datagrams in the presence of subnets J. Mogul
- RFC 927**
TACACS user identification Telnet option B.A. Anderson
- RFC 933**
Output marking Telnet option S. Silverman
- RFC 946**
Telnet terminal location number option R. Nedved
- RFC 950**
Internet Standard Subnetting Procedure J. Mogul, J. Postel
- RFC 952**
DoD Internet host table specification K. Harrenstien, M. Stahl, E. Feinler
- RFC 959**
File Transfer Protocol J. Postel, J.K. Reynolds
- RFC 961**
Official ARPA-Internet protocols J.K. Reynolds, J. Postel
- RFC 974**
Mail routing and the domain system C. Partridge
- RFC 1001**
Protocol standard for a NetBIOS service on a TCP/UDP transport: Concepts and methods NetBios Working Group in the Defense Advanced Research Projects Agency, Internet Activities Board, End-to-End Services Task Force
- RFC 1002**
Protocol Standard for a NetBIOS service on a TCP/UDP transport: Detailed specifications NetBios Working Group in the Defense Advanced Research Projects Agency, Internet Activities Board, End-to-End Services Task Force
- RFC 1006**
ISO transport services on top of the TCP: Version 3 M.T. Rose, D.E. Cass
- RFC 1009**
Requirements for Internet gateways R. Braden, J. Postel
- RFC 1011**
Official Internet protocols J. Reynolds, J. Postel
- RFC 1013**
X Window System Protocol, version 11: Alpha update April 1987 R. Scheifler
- RFC 1014**
XDR: External Data Representation standard Sun Microsystems
- RFC 1027**
Using ARP to implement transparent subnet gateways S. Carl-Mitchell, J. Quarterman

- RFC 1032**
Domain administrators guide M. Stahl
- RFC 1033**
Domain administrators operations guide M. Lottor
- RFC 1034**
Domain names—concepts and facilities P.V. Mockapetris
- RFC 1035**
Domain names—implementation and specification P.V. Mockapetris
- RFC 1038**
Draft revised IP security option M. St. Johns
- RFC 1041**
Telnet 3270 regime option Y. Rekhter
- RFC 1042**
Standard for the transmission of IP datagrams over IEEE 802 networks J. Postel, J. Reynolds
- RFC 1043**
Telnet Data Entry Terminal option: DODIIS implementation A. Yasuda, T. Thompson
- RFC 1044**
Internet Protocol on Network System's HYPERchannel: Protocol specification K. Hardwick, J. Lekashman
- RFC 1053**
Telnet X.3 PAD option S. Levy, T. Jacobson
- RFC 1055**
Nonstandard for transmission of IP datagrams over serial lines: SLIP J. Romkey
- RFC 1057**
RPC: Remote Procedure Call Protocol Specification: Version 2 Sun Microsystems
- RFC 1058**
Routing Information Protocol C. Hedrick
- RFC 1060**
Assigned numbers J. Reynolds, J. Postel
- RFC 1067**
Simple Network Management Protocol J.D. Case, M. Fedor, M.L. Schoffstall, J. Davin
- RFC 1071**
Computing the Internet checksum R.T. Braden, D.A. Borman, C. Partridge
- RFC 1072**
TCP extensions for long-delay paths V. Jacobson, R.T. Braden
- RFC 1073**
Telnet window size option D. Waitzman
- RFC 1079**
Telnet terminal speed option C. Hedrick
- RFC 1085**
ISO presentation services on top of TCP/IP based internets M.T. Rose
- RFC 1091**
Telnet terminal-type option J. VanBokkelen
- RFC 1094**
NFS: Network File System Protocol specification Sun Microsystems
- RFC 1096**
Telnet X display location option G. Marcy
- RFC 1101**
DNS encoding of network names and other types P. Mockapetris

- RFC 1112**
Host extensions for IP multicasting S.E. Deering
- RFC 1113**
Privacy enhancement for Internet electronic mail: Part I — message encipherment and authentication procedures J. Linn
- RFC 1118**
Hitchhikers Guide to the Internet E. Krol
- RFC 1122**
Requirements for Internet Hosts—Communication Layers R. Braden, Ed.
- RFC 1123**
Requirements for Internet Hosts—Application and Support R. Braden, Ed.
- RFC 1146**
TCP alternate checksum options J. Zweig, C. Partridge
- RFC 1155**
Structure and identification of management information for TCP/IP-based internets M. Rose, K. McCloghrie
- RFC 1156**
Management Information Base for network management of TCP/IP-based internets K. McCloghrie, M. Rose
- RFC 1157**
Simple Network Management Protocol (SNMP) J. Case, M. Fedor, M. Schoffstall, J. Davin
- RFC 1158**
Management Information Base for network management of TCP/IP-based internets: MIB-II M. Rose
- RFC 1166**
Internet numbers S. Kirkpatrick, M.K. Stahl, M. Recker
- RFC 1179**
Line printer daemon protocol L. McLaughlin
- RFC 1180**
TCP/IP tutorial T. Socolofsky, C. Kale
- RFC 1183**
New DNS RR Definitions C.F. Everhart, L.A. Mamakos, R. Ullmann, P.V. Mockapetris
- RFC 1184**
Telnet Linemode Option D. Borman
- RFC 1186**
MD4 Message Digest Algorithm R.L. Rivest
- RFC 1187**
Bulk Table Retrieval with the SNMP M. Rose, K. McCloghrie, J. Davin
- RFC 1188**
Proposed Standard for the Transmission of IP Datagrams over FDDI Networks D. Katz
- RFC 1190**
Experimental Internet Stream Protocol: Version 2 (ST-II) C. Topolcic
- RFC 1191**
Path MTU discovery J. Mogul, S. Deering
- RFC 1198**
FYI on the X window system R. Scheifler
- RFC 1207**
FYI on Questions and Answers: Answers to commonly asked “experienced Internet user” questions G. Malkin, A. Marine, J. Reynolds
- RFC 1208**
Glossary of networking terms O. Jacobsen, D. Lynch

RFC 1213

Management Information Base for Network Management of TCP/IP-based internets: MIB-II K. McCloghrie, M.T. Rose

RFC 1215

Convention for defining traps for use with the SNMP M. Rose

RFC 1227

SNMP MUX protocol and MIB M.T. Rose

RFC 1228

SNMP-DPI: Simple Network Management Protocol Distributed Program Interface G. Carpenter, B. Wijnen

RFC 1229

Extensions to the generic-interface MIB K. McCloghrie

RFC 1230

IEEE 802.4 Token Bus MIB K. McCloghrie, R. Fox

RFC 1231

IEEE 802.5 Token Ring MIB K. McCloghrie, R. Fox, E. Decker

RFC 1236

IP to X.121 address mapping for DDN L. Morales, P. Hasse

RFC 1256

ICMP Router Discovery Messages S. Deering, Ed.

RFC 1267

Border Gateway Protocol 3 (BGP-3) K. Lougheed, Y. Rekhter

RFC 1268

Application of the Border Gateway Protocol in the Internet Y. Rekhter, P. Gross

RFC 1269

Definitions of Managed Objects for the Border Gateway Protocol: Version 3 S. Willis, J. Burruss

RFC 1270

SNMP Communications Services F. Kastenholz, ed.

RFC 1285

FDDI Management Information Base J. Case

RFC 1315

Management Information Base for Frame Relay DTEs C. Brown, F. Baker, C. Carvalho

RFC 1321

The MD5 Message-Digest Algorithm R. Rivest

RFC 1323

TCP Extensions for High Performance V. Jacobson, R. Braden, D. Borman

RFC 1325

FYI on Questions and Answers: Answers to Commonly Asked "New Internet User" Questions G. Malkin, A. Marine

RFC 1327

Mapping between X.400 (1988)/ISO 10021 and RFC 822 S. Hardcastle-Kille

RFC 1340

Assigned Numbers J. Reynolds, J. Postel

RFC 1344

Implications of MIME for Internet Mail Gateways N. Bornstein

RFC 1349

Type of Service in the Internet Protocol Suite P. Almquist

RFC 1351

SNMP Administrative Model J. Davin, J. Galvin, K. McCloghrie

- RFC 1352**
SNMP Security Protocols J. Galvin, K. McCloghrie, J. Davin
- RFC 1353**
Definitions of Managed Objects for Administration of SNMP Parties K. McCloghrie, J. Davin, J. Galvin
- RFC 1354**
IP Forwarding Table MIB F. Baker
- RFC 1356**
Multiprotocol Interconnect[®] on X.25 and ISDN in the Packet Mode A. Malis, D. Robinson, R. Ullmann
- RFC 1358**
Charter of the Internet Architecture Board (IAB) L. Chapin
- RFC 1363**
A Proposed Flow Specification C. Partridge
- RFC 1368**
Definition of Managed Objects for IEEE 802.3 Repeater Devices D. McMaster, K. McCloghrie
- RFC 1372**
Telnet Remote Flow Control Option C. L. Hedrick, D. Borman
- RFC 1374**
IP and ARP on HIPPI J. Renwick, A. Nicholson
- RFC 1381**
SNMP MIB Extension for X.25 LAPB D. Throop, F. Baker
- RFC 1382**
SNMP MIB Extension for the X.25 Packet Layer D. Throop
- RFC 1387**
RIP Version 2 Protocol Analysis G. Malkin
- RFC 1388**
RIP Version 2 Carrying Additional Information G. Malkin
- RFC 1389**
RIP Version 2 MIB Extensions G. Malkin, F. Baker
- RFC 1390**
Transmission of IP and ARP over FDDI Networks D. Katz
- RFC 1393**
Traceroute Using an IP Option G. Malkin
- RFC 1398**
Definitions of Managed Objects for the Ethernet-Like Interface Types F. Kastenholz
- RFC 1408**
Telnet Environment Option D. Borman, Ed.
- RFC 1413**
Identification Protocol M. St. Johns
- RFC 1416**
Telnet Authentication Option D. Borman, ed.
- RFC 1420**
SNMP over IPX S. Bostock
- RFC 1428**
Transition of Internet Mail from Just-Send-8 to 8bit-SMTP/MIME G. Vaudreuil
- RFC 1442**
Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser
- RFC 1443**
Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1445

Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2) J. Galvin, K. McCloghrie

RFC 1447

Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2) K. McCloghrie, J. Galvin

RFC 1448

Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1464

Using the Domain Name System to Store Arbitrary String Attributes R. Rosenbaum

RFC 1469

IP Multicast over Token-Ring Local Area Networks T. Pusateri

RFC 1483

Multiprotocol Encapsulation over ATM Adaptation Layer 5 Juha Heinanen

RFC 1514

Host Resources MIB P. Grillo, S. Waldbusser

RFC 1516

Definitions of Managed Objects for IEEE 802.3 Repeater Devices D. McMaster, K. McCloghrie

RFC 1521

MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies N. Borenstein, N. Freed

RFC 1535

A Security Problem and Proposed Correction With Widely Deployed DNS Software E. Gavron

RFC 1536

Common DNS Implementation Errors and Suggested Fixes A. Kumar, J. Postel, C. Neuman, P. Danzig, S. Miller

RFC 1537

Common DNS Data File Configuration Errors P. Beertema

RFC 1540

Internet Official Protocol Standards J. Postel

RFC 1571

Telnet Environment Option Interoperability Issues D. Borman

RFC 1572

Telnet Environment Option S. Alexander

RFC 1573

Evolution of the Interfaces Group of MIB-II K. McCloghrie, F. Kastenholz

RFC 1577

Classical IP and ARP over ATM M. Laubach

RFC 1583

OSPF Version 2 J. Moy

RFC 1591

Domain Name System Structure and Delegation J. Postel

RFC 1592

Simple Network Management Protocol Distributed Protocol Interface Version 2.0 B. Wijnen, G. Carpenter, K. Curran, A. Sehgal, G. Waters

RFC 1594

FYI on Questions and Answers—Answers to Commonly Asked "New Internet User" Questions A. Marine, J. Reynolds, G. Malkin

RFC 1644

T/TCP — TCP Extensions for Transactions Functional Specification R. Braden

- RFC 1646**
TN3270 Extensions for LUsername and Printer Selection C. Graves, T. Butts, M. Angel
- RFC 1647**
TN3270 Enhancements B. Kelly
- RFC 1652**
SMTP Service Extension for 8bit-MIMEtransport J. Klensin, N. Freed, M. Rose, E. Stefferud, D. Crocker
- RFC 1664**
Using the Internet DNS to Distribute RFC1327 Mail Address Mapping Tables C. Allochio, A. Bonito, B. Cole, S. Giordano, R. Hagens
- RFC 1693**
An Extension to TCP: Partial Order Service T. Connolly, P. Amer, P. Conrad
- RFC 1695**
Definitions of Managed Objects for ATM Management Version 8.0 using SMIPv2 M. Ahmed, K. Tesink
- RFC 1701**
Generic Routing Encapsulation (GRE) S. Hanks, T. Li, D. Farinacci, P. Traina
- RFC 1702**
Generic Routing Encapsulation over IPv4 networks S. Hanks, T. Li, D. Farinacci, P. Traina
- RFC 1706**
DNS NSAP Resource Records B. Manning, R. Colella
- RFC 1712**
DNS Encoding of Geographical Location C. Farrell, M. Schulze, S. Pleitner D. Baldoni
- RFC 1713**
Tools for DNS debugging A. Romao
- RFC 1723**
RIP Version 2—Carrying Additional Information G. Malkin
- RFC 1752**
The Recommendation for the IP Next Generation Protocol S. Bradner, A. Mankin
- RFC 1766**
Tags for the Identification of Languages H. Alvestrand
- RFC 1771**
A Border Gateway Protocol 4 (BGP-4) Y. Rekhter, T. Li
- RFC 1794**
DNS Support for Load Balancing T. Brisco
- RFC 1819**
Internet Stream Protocol Version 2 (ST2) Protocol Specification—Version ST2+ L. Delgrossi, L. Berger Eds.
- RFC 1826**
IP Authentication Header R. Atkinson
- RFC 1828**
IP Authentication using Keyed MD5 P. Metzger, W. Simpson
- RFC 1829**
The ESP DES-CBC Transform P. Karn, P. Metzger, W. Simpson
- RFC 1830**
SMTP Service Extensions for Transmission of Large and Binary MIME Messages G. Vaudreuil
- RFC 1831**
RPC: Remote Procedure Call Protocol Specification Version 2 R. Srinivasan
- RFC 1832**
XDR: External Data Representation Standard R. Srinivasan
- RFC 1833**
Binding Protocols for ONC RPC Version 2 R. Srinivasan

RFC 1850

OSPF Version 2 Management Information Base F. Baker, R. Coltun

RFC 1854

SMTP Service Extension for Command Pipelining N. Freed

RFC 1869

SMTP Service Extensions J. Klensin, N. Freed, M. Rose, E. Stefferud, D. Crocker

RFC 1870

SMTP Service Extension for Message Size Declaration J. Klensin, N. Freed, K. Moore

RFC 1876

A Means for Expressing Location Information in the Domain Name System C. Davis, P. Vixie, T. Goodwin, I. Dickinson

RFC 1883

Internet Protocol, Version 6 (IPv6) Specification S. Deering, R. Hinden

RFC 1884

IP Version 6 Addressing Architecture R. Hinden, S. Deering, Eds.

RFC 1886

DNS Extensions to support IP version 6 S. Thomson, C. Huitema

RFC 1888

OSI NSAPs and IPv6 J. Bound, B. Carpenter, D. Harrington, J. Houldsworth, A. Lloyd

RFC 1891

SMTP Service Extension for Delivery Status Notifications K. Moore

RFC 1892

The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages G. Vaudreuil

RFC 1894

An Extensible Message Format for Delivery Status Notifications K. Moore, G. Vaudreuil

RFC 1901

Introduction to Community-based SNMPv2 J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1902

Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1903

Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1904

Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1905

Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1906

Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1907

Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2) J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1908

Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 1912

Common DNS Operational and Configuration Errors D. Barr

- RFC 1918**
Address Allocation for Private Internets Y. Rekhter, B. Moskowitz, D. Karrenberg, G.J. de Groot, E. Lear
- RFC 1928**
SOCKS Protocol Version 5 M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, L. Jones
- RFC 1930**
Guidelines for creation, selection, and registration of an Autonomous System (AS) J. Hawkinson, T. Bates
- RFC 1939**
Post Office Protocol-Version 3 J. Myers, M. Rose
- RFC 1981**
Path MTU Discovery for IP version 6 J. McCann, S. Deering, J. Mogul
- RFC 1982**
Serial Number Arithmetic R. Elz, R. Bush
- RFC 1985**
SMTP Service Extension for Remote Message Queue Starting J. De Winter
- RFC 1995**
Incremental Zone Transfer in DNS M. Ohta
- RFC 1996**
A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY) P. Vixie
- RFC 2010**
Operational Criteria for Root Name Servers B. Manning, P. Vixie
- RFC 2011**
SNMPv2 Management Information Base for the Internet Protocol using SMIPv2 K. McCloghrie, Ed.
- RFC 2012**
SNMPv2 Management Information Base for the Transmission Control Protocol using SMIPv2 K. McCloghrie, Ed.
- RFC 2013**
SNMPv2 Management Information Base for the User Datagram Protocol using SMIPv2 K. McCloghrie, Ed.
- RFC 2018**
TCP Selective Acknowledgement Options M. Mathis, J. Mahdavi, S. Floyd, A. Romanow
- RFC 2026**
The Internet Standards Process — Revision 3 S. Bradner
- RFC 2030**
Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI D. Mills
- RFC 2033**
Local Mail Transfer Protocol J. Myers
- RFC 2034**
SMTP Service Extension for Returning Enhanced Error Codes N. Freed
- RFC 2040**
The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms R. Baldwin, R. Rivest
- RFC 2045**
Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies N. Freed, N. Borenstein
- RFC 2052**
A DNS RR for specifying the location of services (DNS SRV) A. Gulbrandsen, P. Vixie
- RFC 2065**
Domain Name System Security Extensions D. Eastlake 3rd, C. Kaufman
- RFC 2066**
TELNET CHARSET Option R. Gellens

RFC 2080

RIPng for IPv6 G. Malkin, R. Minnear

RFC 2096

IP Forwarding Table MIB F. Baker

RFC 2104

HMAC: Keyed-Hashing for Message Authentication H. Krawczyk, M. Bellare, R. Canetti

RFC 2119

Keywords for use in RFCs to Indicate Requirement Levels S. Bradner

RFC 2133

Basic Socket Interface Extensions for IPv6 R. Gilligan, S. Thomson, J. Bound, W. Stevens

RFC 2136

Dynamic Updates in the Domain Name System (DNS UPDATE) P. Vixie, Ed., S. Thomson, Y. Rekhter, J. Bound

RFC 2137

Secure Domain Name System Dynamic Update D. Eastlake 3rd

RFC 2163

Using the Internet DNS to Distribute MIXER Conformant Global Address Mapping (MCGAM) C. Allocchio

RFC 2168

Resolution of Uniform Resource Identifiers using the Domain Name System R. Daniel, M. Mealling

RFC 2178

OSPF Version 2 J. Moy

RFC 2181

Clarifications to the DNS Specification R. Elz, R. Bush

RFC 2205

Resource ReSerVation Protocol (RSVP)—Version 1 Functional Specification R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin

RFC 2210

The Use of RSVP with IETF Integrated Services J. Wroclawski

RFC 2211

Specification of the Controlled-Load Network Element Service J. Wroclawski

RFC 2212

Specification of Guaranteed Quality of Service S. Shenker, C. Partridge, R. Guerin

RFC 2215

General Characterization Parameters for Integrated Service Network Elements S. Shenker, J. Wroclawski

RFC 2217

Telnet Com Port Control Option G. Clarke

RFC 2219

Use of DNS Aliases for Network Services M. Hamilton, R. Wright

RFC 2228

FTP Security Extensions M. Horowitz, S. Lunt

RFC 2230

Key Exchange Delegation Record for the DNS R. Atkinson

RFC 2233

The Interfaces Group MIB using SMIV2 K. McCloghrie, F. Kastenholz

RFC 2240

A Legal Basis for Domain Name Allocation O. Vaughn

RFC 2246

The TLS Protocol Version 1.0 T. Dierks, C. Allen

RFC 2251

Lightweight Directory Access Protocol (v3) M. Wahl, T. Howes, S. Kille

RFC 2253

Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names M. Wahl, S. Kille, T. Howes

RFC 2254

The String Representation of LDAP Search Filters T. Howes

RFC 2261

An Architecture for Describing SNMP Management Frameworks D. Harrington, R. Presuhn, B. Wijnen

RFC 2262

Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) J. Case, D. Harrington, R. Presuhn, B. Wijnen

RFC 2271

An Architecture for Describing SNMP Management Frameworks D. Harrington, R. Presuhn, B. Wijnen

RFC 2273

SNMPv3 Applications D. Levi, P. Meyer, B. Stewartz

RFC 2274

User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3) U. Blumenthal, B. Wijnen

RFC 2275

View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP) B. Wijnen, R. Presuhn, K. McCloghrie

RFC 2279

UTF-8, a transformation format of ISO 10646 F. Yergeau

RFC 2292

Advanced Sockets API for IPv6 W. Stevens, M. Thomas

RFC 2308

Negative Caching of DNS Queries (DNS NCACHE) M. Andrews

RFC 2317

Classless IN-ADDR.ARPA delegation H. Eidnes, G. de Groot, P. Vixie

RFC 2320

Definitions of Managed Objects for Classical IP and ARP Over ATM Using SMIPv2 (IPOA-MIB) M. Greene, J. Luciani, K. White, T. Kuo

RFC 2328

OSPF Version 2 J. Moy

RFC 2345

Domain Names and Company Name Retrieval J. Klensin, T. Wolf, G. Oglesby

RFC 2352

A Convention for Using Legal Names as Domain Names O. Vaughn

RFC 2355

TN3270 Enhancements B. Kelly

RFC 2358

Definitions of Managed Objects for the Ethernet-like Interface Types J. Flick, J. Johnson

RFC 2373

IP Version 6 Addressing Architecture R. Hinden, S. Deering

RFC 2374

An IPv6 Aggregatable Global Unicast Address Format R. Hinden, M. O'Dell, S. Deering

RFC 2375

IPv6 Multicast Address Assignments R. Hinden, S. Deering

RFC 2385

Protection of BGP Sessions via the TCP MD5 Signature Option A. Hefferman

RFC 2389

Feature negotiation mechanism for the File Transfer Protocol P. Hethmon, R. Elz

RFC 2401

Security Architecture for Internet Protocol S. Kent, R. Atkinson

RFC 2402

IP Authentication Header S. Kent, R. Atkinson

RFC 2403

The Use of HMAC-MD5-96 within ESP and AH C. Madson, R. Glenn

RFC 2404

The Use of HMAC-SHA-1-96 within ESP and AH C. Madson, R. Glenn

RFC 2405

The ESP DES-CBC Cipher Algorithm With Explicit IV C. Madson, N. Doraswamy

RFC 2406

IP Encapsulating Security Payload (ESP) S. Kent, R. Atkinson

RFC 2407

The Internet IP Security Domain of Interpretation for ISAKMPD Piper

RFC 2408

Internet Security Association and Key Management Protocol (ISAKMP) D. Maughan, M. Schertler, M. Schneider, J. Turner

RFC 2409

The Internet Key Exchange (IKE) D. Harkins, D. Carrel

RFC 2410

The NULL Encryption Algorithm and Its Use With IPsec R. Glenn, S. Kent,

RFC 2428

FTP Extensions for IPv6 and NATs M. Allman, S. Ostermann, C. Metz

RFC 2445

Internet Calendaring and Scheduling Core Object Specification (iCalendar) F. Dawson, D. Stenerson

RFC 2459

Internet X.509 Public Key Infrastructure Certificate and CRL Profile R. Housley, W. Ford, W. Polk, D. Solo

RFC 2460

Internet Protocol, Version 6 (IPv6) Specification S. Deering, R. Hinden

RFC 2461

Neighbor Discovery for IP Version 6 (IPv6) T. Narten, E. Nordmark, W. Simpson

RFC 2462

IPv6 Stateless Address Autoconfiguration S. Thomson, T. Narten

RFC 2463

Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification A. Conta, S. Deering

RFC 2464

Transmission of IPv6 Packets over Ethernet Networks M. Crawford

RFC 2466

Management Information Base for IP Version 6: ICMPv6 Group D. Haskin, S. Onishi

RFC 2476

Message Submission R. Gellens, J. Klensin

RFC 2487

SMTP Service Extension for Secure SMTP over TLS P. Hoffman

RFC 2505

Anti-Spam Recommendations for SMTP MTAs G. Lindberg

- RFC 2523**
Photuris: Extended Schemes and Attributes P. Karn, W. Simpson
- RFC 2535**
Domain Name System Security Extensions D. Eastlake 3rd
- RFC 2538**
Storing Certificates in the Domain Name System (DNS) D. Eastlake 3rd, O. Gudmundsson
- RFC 2539**
Storage of Diffie-Hellman Keys in the Domain Name System (DNS) D. Eastlake 3rd
- RFC 2540**
Detached Domain Name System (DNS) Information D. Eastlake 3rd
- RFC 2554**
SMTP Service Extension for Authentication J. Myers
- RFC 2570**
Introduction to Version 3 of the Internet-standard Network Management Framework J. Case, R. Mundy, D. Partain, B. Stewart
- RFC 2571**
An Architecture for Describing SNMP Management Frameworks B. Wijnen, D. Harrington, R. Presuhn
- RFC 2572**
Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) J. Case, D. Harrington, R. Presuhn, B. Wijnen
- RFC 2573**
SNMP Applications D. Levi, P. Meyer, B. Stewart
- RFC 2574**
User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3) U. Blumenthal, B. Wijnen
- RFC 2575**
View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP) B. Wijnen, R. Presuhn, K. McCloghrie
- RFC 2576**
Co-Existence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework R. Frye, D. Levi, S. Routhier, B. Wijnen
- RFC 2578**
Structure of Management Information Version 2 (SMIv2) K. McCloghrie, D. Perkins, J. Schoenwaelder
- RFC 2579**
Textual Conventions for SMIv2 K. McCloghrie, D. Perkins, J. Schoenwaelder
- RFC 2580**
Conformance Statements for SMIv2 K. McCloghrie, D. Perkins, J. Schoenwaelder
- RFC 2581**
TCP Congestion Control M. Allman, V. Paxson, W. Stevens
- RFC 2583**
Guidelines for Next Hop Client (NHC) Developers R. Carlson, L. Winkler
- RFC 2591**
Definitions of Managed Objects for Scheduling Management Operations D. Levi, J. Schoenwaelder
- RFC 2625**
IP and ARP over Fibre Channel M. Rajagopal, R. Bhagwat, W. Rickard
- RFC 2635**
Don't SPEW A Set of Guidelines for Mass Unsolicited Mailings and Postings (spam)* S. Hambridge, A. Lunde
- RFC 2637**
Point-to-Point Tunneling Protocol K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, G. Zorn

- RFC 2640**
Internationalization of the File Transfer Protocol B. Curtin
- RFC 2665**
Definitions of Managed Objects for the Ethernet-like Interface Types J. Flick, J. Johnson
- RFC 2671**
Extension Mechanisms for DNS (EDNS0) P. Vixie
- RFC 2672**
Non-Terminal DNS Name Redirection M. Crawford
- RFC 2675**
IPv6 Jumbograms D. Borman, S. Deering, R. Hinden
- RFC 2710**
Multicast Listener Discovery (MLD) for IPv6 S. Deering, W. Fenner, B. Haberman
- RFC 2711**
IPv6 Router Alert Option C. Partridge, A. Jackson
- RFC 2740**
OSPF for IPv6 R. Coltun, D. Ferguson, J. Moy
- RFC 2753**
A Framework for Policy-based Admission Control R. Yavatkar, D. Pendarakis, R. Guerin
- RFC 2782**
A DNS RR for specifying the location of services (DNS SRV) A. Gubbrandsen, P. Vixie, L. Esibov
- RFC 2821**
Simple Mail Transfer Protocol J. Klensin, Ed.
- RFC 2822**
Internet Message Format P. Resnick, Ed.
- RFC 2840**
TELNET KERMIT OPTION J. Altman, F. da Cruz
- RFC 2845**
Secret Key Transaction Authentication for DNS (TSIG) P. Vixie, O. Gudmundsson, D. Eastlake 3rd, B. Wellington
- RFC 2851**
Textual Conventions for Internet Network Addresses M. Daniele, B. Haberman, S. Routhier, J. Schoenwaelder
- RFC 2852**
Deliver By SMTP Service Extension D. Newman
- RFC 2874**
DNS Extensions to Support IPv6 Address Aggregation and Renumbering M. Crawford, C. Huitema
- RFC 2915**
The Naming Authority Pointer (NAPTR) DNS Resource Record M. Mealling, R. Daniel
- RFC 2920**
SMTP Service Extension for Command Pipelining N. Freed
- RFC 2930**
Secret Key Establishment for DNS (TKEY RR) D. Eastlake, 3rd
- RFC 2941**
Telnet Authentication Option T. Ts'o, ed., J. Altman
- RFC 2942**
Telnet Authentication: Kerberos Version 5 T. Ts'o
- RFC 2946**
Telnet Data Encryption Option T. Ts'o
- RFC 2952**
Telnet Encryption: DES 64 bit Cipher Feedback T. Ts'o

RFC 2953

Telnet Encryption: DES 64 bit Output Feedback T. Ts'o

RFC 2992

Analysis of an Equal-Cost Multi-Path Algorithm C. Hopps

RFC 3019

IP Version 6 Management Information Base for The Multicast Listener Discovery Protocol B. Haberman, R. Worzella

RFC 3060

Policy Core Information Model—Version 1 Specification B. Moore, E. Ellessen, J. Strassner, A. Westerinen

RFC 3152

Delegation of IP6.ARPA R. Bush

RFC 3164

The BSD Syslog Protocol C. Lonvick

RFC 3207

SMTP Service Extension for Secure SMTP over Transport Layer Security P. Hoffman

RFC 3226

DNSSEC and IPv6 A6 aware server/resolver message size requirements O. Gudmundsson

RFC 3291

Textual Conventions for Internet Network Addresses M. Daniele, B. Haberman, S. Routhier, J. Schoenwaelder

RFC 3363

Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System R. Bush, A. Durand, B. Fink, O. Gudmundsson, T. Hain

RFC 3376

Internet Group Management Protocol, Version 3 B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan

RFC 3390

Increasing TCP's Initial Window M. Allman, S. Floyd, C. Partridge

RFC 3410

Introduction and Applicability Statements for Internet-Standard Management Framework J. Case, R. Mundy, D. Partain, B. Stewart

RFC 3411

An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks D. Harrington, R. Presuhn, B. Wijnen

RFC 3412

Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) J. Case, D. Harrington, R. Presuhn, B. Wijnen

RFC 3413

Simple Network Management Protocol (SNMP) Applications D. Levi, P. Meyer, B. Stewart

RFC 3414

User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3) U. Blumenthal, B. Wijnen

RFC 3415

View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP) B. Wijnen, R. Presuhn, K. McCloghrie

RFC 3416

Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP) R. Presuhn, J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 3417

Transport Mappings for the Simple Network Management Protocol (SNMP) R. Presuhn, J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 3418

Management Information Base (MIB) for the Simple Network Management Protocol (SNMP) R. Presuhn, J. Case, K. McCloghrie, M. Rose, S. Waldbusser

RFC 3419

Textual Conventions for Transport Addresses M. Daniele, J. Schoenwaelder

RFC 3484

Default Address Selection for Internet Protocol version 6 (IPv6) R. Draves

RFC 3493

Basic Socket Interface Extensions for IPv6 R. Gilligan, S. Thomson, J. Bound, J. McCann, W. Stevens

RFC 3513

Internet Protocol Version 6 (IPv6) Addressing Architecture R. Hinden, S. Deering

RFC 3526

More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE) T. Kivinen, M. Kojo

RFC 3542

Advanced Sockets Application Programming Interface (API) for IPv6 W. Richard Stevens, M. Thomas, E. Nordmark, T. Jinmei

RFC 3566

The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec S. Frankel, H. Herbert

RFC 3569

An Overview of Source-Specific Multicast (SSM) S. Bhattacharyya, Ed.

RFC 3584

Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework R. Frye, D. Levi, S. Routhier, B. Wijnen

RFC 3602

The AES-CBC Cipher Algorithm and Its Use with IPsec S. Frankel, R. Glenn, S. Kelly

RFC 3629

UTF-8, a transformation format of ISO 10646 R. Kermode, C. Vicisano

RFC 3658

Delegation Signer (DS) Resource Record (RR) O. Gudmundsson

RFC 3678

Socket Interface Extensions for Multicast Source Filters D. Thaler, B. Fenner, B. Quinn

RFC 3715

IPsec-Network Address Translation (NAT) Compatibility Requirements B. Aboba, W. Dixon

RFC 3810

Multicast Listener Discovery Version 2 (MLDv2) for IPv6 R. Vida, Ed., L. Costa, Ed.

RFC 3826

The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model U. Blumenthal, F. Maino, K. McCloghrie.

RFC 3947

Negotiation of NAT-Traversal in the IKE T. Kivinen, B. Swander, A. Huttunen, V. Volpe

RFC 3948

UDP Encapsulation of IPsec ESP Packets A. Huttunen, B. Swander, V. Volpe, L. DiBurro, M. Stenberg

RFC 4001

Textual Conventions for Internet Network Addresses M. Daniele, B. Haberman, S. Routhier, J. Schoenwaelder

RFC 4007

IPv6 Scoped Address Architecture S. Deering, B. Haberman, T. Jinmei, E. Nordmark, B. Zill

- RFC 4022**
Management Information Base for the Transmission Control Protocol (TCP) R. Raghunarayan
- RFC 4106**
The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP) J. Viega, D. McGrew
- RFC 4109**
Algorithms for Internet Key Exchange version 1 (IKEv1) P. Hoffman
- RFC 4113**
Management Information Base for the User Datagram Protocol (UDP) B. Fenner, J. Flick
- RFC 4191**
Default Router Preferences and More-Specific Routes R. Draves, D. Thaler
- RFC 4217**
Securing FTP with TLS P. Ford-Hutchinson
- RFC 4292**
IP Forwarding Table MIB B. Haberman
- RFC 4293**
Management Information Base for the Internet Protocol (IP) S. Routhier
- RFC 4301**
Security Architecture for the Internet Protocol S. Kent, K. Seo
- RFC 4302**
IP Authentication Header S. Kent
- RFC 4303**
IP Encapsulating Security Payload (ESP) S. Kent
- RFC 4304**
Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP) S. Kent
- RFC 4307**
Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2) J. Schiller
- RFC 4308**
Cryptographic Suites for IPsec P. Hoffman
- RFC 4434**
The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol P. Hoffman
- RFC 4443**
Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification A. Conta, S. Deering
- RFC 4552**
Authentication/Confidentiality for OSPFv3 M. Gupta, N. Melam
- RFC 4678**
Server/Application State Protocol v1 A. Bivens
- RFC 4753**
ECP Groups for IKE and IKEv2 D. Fu, J. Solinas
- RFC 4754**
IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA) D. Fu, J. Solinas
- RFC 4809**
Requirements for an IPsec Certificate Management Profile C. Bonatti, Ed., S. Turner, Ed., G. Lebovitz, Ed.
- RFC 4835**
Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH) V. Manral

RFC 4862

IPv6 Stateless Address Autoconfiguration S. Thomson, T. Narten, T. Jinmei

RFC 4868

Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec S. Kelly, S. Frankel

RFC 4869

Suite B Cryptographic Suites for IPsec L. Law, J. Solinas

RFC 4941

Privacy Extensions for Stateless Address Autoconfiguration in IPv6 T. Narten, R. Draves, S. Krishnan

RFC 4945

The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2, and PKIX B. Korver

RFC 5014

IPv6 Socket API for Source Address Selection E. Nordmark, S. Chakrabarti, J. Laganier

RFC 5095

Deprecation of Type 0 Routing Headers in IPv6 J. Abley, P. Savola, G. Neville-Neil

RFC 5175

IPv6 Router Advertisement Flags Option B. Haberman, Ed., R. Hinden

RFC 5282

Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol D. Black, D. McGrew

RFC 5996

Internet Key Exchange Protocol Version 2 (IKEv2) C. Kaufman, P. Hoffman, Y. Nir, P. Eronen

RFC 7627

Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension K. Bhargavan, A. Delignat-Lavaud, A. Pironti, Inria Paris-Rocquencourt, A. Langley, M. Ray

RFC 8446

The Transport Layer Security (TLS) Protocol Version 1.3 E. Rescorla

Internet drafts

Internet drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Other groups can also distribute working documents as Internet drafts. You can see Internet drafts at <http://www.ietf.org/ID.html>.

Appendix E. Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you can view the information through the z/OS Internet Library website <http://www.ibm.com/systems/z/os/zos/library/bkserv/> or IBM Documentation <https://www.ibm.com/docs/en>. If you continue to experience problems, send a message to Contact z/OS web page (www.ibm.com/systems/z/os/zos/webqs.html) or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. See *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

One exception is command syntax that is published in railroad track format, which is accessible using screen readers with IBM Documentation, as described in [“Dotted decimal syntax diagrams” on page 401](#).

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing IBM Documentation using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should see separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- A question mark (?) means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- An exclamation mark (!) means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP applies only to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.
- An asterisk (*) means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

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 the 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 DFSMS, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [IBM Lifecycle Support for z/OS \(www.ibm.com/software/support/systemsz/lifecycle\)](http://www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

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 at www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Bibliography

This bibliography contains descriptions of the documents in the z/OS Communications Server library.

z/OS Communications Server documentation is available online at the z/OS Internet Library web page at <http://www.ibm.com/systems/z/os/zos/library/bkserv/>.

z/OS Communications Server library updates

Updates to documents are also available on RETAIN and in information APARs (info APARs). Go to <https://www.ibm.com/mysupport> to view information APARs.

- [z/OS Communications Server V2R1 New Function APAR Summary](#)
- [z/OS Communications Server V2R2 New Function APAR Summary](#)
- [z/OS Communications Server V2R3 New Function APAR Summary](#)
- [z/OS Communications Server V2R4 New Function APAR Summary](#)

z/OS Communications Server information

z/OS Communications Server product information is grouped by task in the following tables.

Planning

| Title | Number | Description |
|---|-----------|---|
| z/OS Communications Server: New Function Summary | GC27-3664 | This document is intended to help you plan for new IP or SNA functions, whether you are migrating from a previous version or installing z/OS for the first time. It summarizes what is new in the release and identifies the suggested and required modifications needed to use the enhanced functions. |
| z/OS Communications Server: IPv6 Network and Application Design Guide | SC27-3663 | This document is a high-level introduction to IPv6. It describes concepts of z/OS Communications Server's support of IPv6, coexistence with IPv4, and migration issues. |

Resource definition, configuration, and tuning

| Title | Number | Description |
|--|-----------|--|
| z/OS Communications Server: IP Configuration Guide | SC27-3650 | This document describes the major concepts involved in understanding and configuring an IP network. Familiarity with the z/OS operating system, IP protocols, z/OS UNIX System Services, and IBM Time Sharing Option (TSO) is recommended. Use this document with the z/OS Communications Server: IP Configuration Reference . |

| Title | Number | Description |
|---|---------------|---|
| z/OS Communications Server: IP Configuration Reference | SC27-3651 | This document presents information for people who want to administer and maintain IP. Use this document with the z/OS Communications Server: IP Configuration Guide . The information in this document includes: <ul style="list-style-type: none"> • TCP/IP configuration data sets • Configuration statements • Translation tables • Protocol number and port assignments |
| z/OS Communications Server: SNA Network Implementation Guide | SC27-3672 | This document presents the major concepts involved in implementing an SNA network. Use this document with the z/OS Communications Server: SNA Resource Definition Reference . |
| z/OS Communications Server: SNA Resource Definition Reference | SC27-3675 | This document describes each SNA definition statement, start option, and macroinstruction for user tables. It also describes NCP definition statements that affect SNA. Use this document with the z/OS Communications Server: SNA Network Implementation Guide . |
| z/OS Communications Server: SNA Resource Definition Samples | SC27-3676 | This document contains sample definitions to help you implement SNA functions in your networks, and includes sample major node definitions. |
| z/OS Communications Server: IP Network Print Facility | SC27-3658 | This document is for systems programmers and network administrators who need to prepare their network to route SNA, JES2, or JES3 printer output to remote printers using TCP/IP Services. |

Operation

| Title | Number | Description |
|--|---------------|---|
| z/OS Communications Server: IP User's Guide and Commands | SC27-3662 | This document describes how to use TCP/IP applications. It contains requests with which a user can log on to a remote host using Telnet, transfer data sets using FTP, send electronic mail, print on remote printers, and authenticate network users. |
| z/OS Communications Server: IP System Administrator's Commands | SC27-3661 | This document describes the functions and commands helpful in configuring or monitoring your system. It contains system administrator's commands, such as TSO NETSTAT, PING, TRACERTE and their UNIX counterparts. It also includes TSO and MVS commands commonly used during the IP configuration process. |
| z/OS Communications Server: SNA Operation | SC27-3673 | This document serves as a reference for programmers and operators requiring detailed information about specific operator commands. |
| z/OS Communications Server: Quick Reference | SC27-3665 | This document contains essential information about SNA and IP commands. |

Customization

| Title | Number | Description |
|---|-----------|---|
| z/OS Communications Server: SNA Customization | SC27-3666 | <p>This document enables you to customize SNA, and includes the following information:</p> <ul style="list-style-type: none"> • Communication network management (CNM) routing table • Logon-interpret routine requirements • Logon manager installation-wide exit routine for the CLU search exit • TSO/SNA installation-wide exit routines • SNA installation-wide exit routines |

Writing application programs

| Title | Number | Description |
|--|-----------|--|
| z/OS Communications Server: IP Sockets Application Programming Interface Guide and Reference | SC27-3660 | This document describes the syntax and semantics of program source code necessary to write your own application programming interface (API) into TCP/IP. You can use this interface as the communication base for writing your own client or server application. You can also use this document to adapt your existing applications to communicate with each other using sockets over TCP/IP. |
| z/OS Communications Server: IP CICS Sockets Guide | SC27-3649 | This document is for programmers who want to set up, write application programs for, and diagnose problems with the socket interface for CICS® using z/OS TCP/IP. |
| z/OS Communications Server: IP IMS Sockets Guide | SC27-3653 | This document is for programmers who want application programs that use the IMS TCP/IP application development services provided by the TCP/IP Services of IBM. |
| z/OS Communications Server: IP Programmer's Guide and Reference | SC27-3659 | This document describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing. Familiarity with the z/OS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended. |
| z/OS Communications Server: SNA Programming | SC27-3674 | This document describes how to use SNA macroinstructions to send data to and receive data from (1) a terminal in either the same or a different domain, or (2) another application program in either the same or a different domain. |
| z/OS Communications Server: SNA Programmer's LU 6.2 Guide | SC27-3669 | This document describes how to use the SNA LU 6.2 application programming interface for host application programs. This document applies to programs that use only LU 6.2 sessions or that use LU 6.2 sessions along with other session types. (Only LU 6.2 sessions are covered in this document.) |
| z/OS Communications Server: SNA Programmer's LU 6.2 Reference | SC27-3670 | This document provides reference material for the SNA LU 6.2 programming interface for host application programs. |

| Title | Number | Description |
|---|-----------|--|
| z/OS Communications Server: CSM Guide | SC27-3647 | This document describes how applications use the communications storage manager. |

Diagnosis

| Title | Number | Description |
|--|------------------------|--|
| z/OS Communications Server: IP Diagnosis Guide | GC27-3652 | This document explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the TCP/IP product code. It explains how to gather information for and describe problems to the IBM Software Support Center. |
| z/OS Communications Server: ACF/TAP Trace Analysis Handbook | GC27-3645 | This document explains how to gather the trace data that is collected and stored in the host processor. It also explains how to use the Advanced Communications Function/Trace Analysis Program (ACF/TAP) service aid to produce reports for analyzing the trace data information. |
| z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures and z/OS Communications Server: SNA Diagnosis Vol 2, FFST Dumps and the VIT | GC27-3667 GC27-3668 | These documents help you identify an SNA problem, classify it, and collect information about it before you call the IBM Support Center. The information collected includes traces, dumps, and other problem documentation. |
| z/OS Communications Server: SNA Data Areas Volume 1 and z/OS Communications Server: SNA Data Areas Volume 2 | GC31-6852 GC31-6853 | These documents describe SNA data areas and can be used to read an SNA dump. They are intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA. |

Messages and codes

| Title | Number | Description |
|---|-----------|---|
| z/OS Communications Server: SNA Messages | SC27-3671 | This document describes the ELM, IKT, IST, IUT, IVT, and USS messages. Other information in this document includes: <ul style="list-style-type: none"> • Command and RU types in SNA messages • Node and ID types in SNA messages • Supplemental message-related information |
| z/OS Communications Server: IP Messages Volume 1 (EZA) | SC27-3654 | This volume contains TCP/IP messages beginning with EZA. |
| z/OS Communications Server: IP Messages Volume 2 (EZB, EZD) | SC27-3655 | This volume contains TCP/IP messages beginning with EZB or EZD. |
| z/OS Communications Server: IP Messages Volume 3 (EZY) | SC27-3656 | This volume contains TCP/IP messages beginning with EYZ. |
| z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM) | SC27-3657 | This volume contains TCP/IP messages beginning with EZZ and SNM. |
| z/OS Communications Server: IP and SNA Codes | SC27-3648 | This document describes codes and other information that appear in z/OS Communications Server messages. |

Index

Special Characters

- d (RSH parameter) [353](#)
- l (RSH parameter) [353](#)
- s port (RSH parameter) [353](#)
- ! (FTP subcommand) [143](#)

Numerics

- 3270 DBCS Transform mode conversion types
 - DECKANJI [17](#)
 - EUCKANJI [17](#)
 - HANGEUL [17](#)
 - JIS78KJ [17](#)
 - JIS83KJ [17](#)
 - KSC5601 [17](#)
 - SJISKANJI [17](#)
 - TCHINESE [17](#)
- 3270 DBCS Transform mode description [16](#)
- 3270 DBCS Transform mode terminal types
 - TTY [17](#)
 - VT100 [17](#)
 - VT282 [17](#)
- 802.3 network protocol [1](#)

A

- A (FTP TYPE parameter) [289](#)
- accessibility [401](#)
- account_information (FTP ACCT parameter) [144](#)
- ACCT (FTP subcommand) [144](#)
- addresses
 - class A network [5](#)
 - description [2](#)
 - loopback [5](#)
 - TCP/IP (idx4) [2](#)
- AIX files [362](#)
- all
 - FTP HELP parameter [171](#)
 - LPQ parameter [316](#)
- allocating data sets
 - FTP input and output data sets [27](#)
 - new data sets [82](#)
- alphanumeric cursor (GDDM/MVS) [334](#)
- ANFontn (GDDMXD option) [337](#)
- AO (TELNET subcommand) [10](#)
- APL2 Character Set keyboard [345](#), [365](#)
- APPEND (FTP subcommand) [144](#)
- appending a local data set to a remote host [144](#)
- applications, functions, and protocols
 - File Transfer Protocol (FTP) [19](#), [43](#)
 - Graphical Data Display Manager Interface for X Windows (GDDMXD) [333](#)
 - Remote Printing (LPR and LPD) [315](#)
 - Telnet [7](#)
 - z/OS UNIX Remote Execution Protocol (orexec) [347](#), [355](#)

- AS/400 files [363](#)
- ASAtans parameter [252](#)
- ASCII
 - ASCII control characters [14](#)
 - FTP JIS78KJ parameter [172](#)
 - FTP JIS83KJ parameter [173](#), [174](#)
 - FTP subcommand [146](#)
 - transferring binary data to EBCDIC [289](#)
 - transferring text data to EBCDIC [289](#)
- AT host
 - LPQ parameter [316](#)
 - LPR parameter [319](#)
 - LPRM parameter [329](#)
- attached graphics cursor (GDDM/MVS) [334](#)
- AUTH (FTP subcommand) [146](#)
- authorizations for data and programs [5](#)
- AUTOMOUNT
 - FTP LOCSITE and SITE parameter [183](#), [252](#)
- AUTORECALL
 - FTP LOCSITE and SITE parameter [183](#), [252](#)
- AYT (TELNET subcommand) [11](#)

B

- B parameter
 - FTP MODE [220](#)
 - FTP TYPE [289](#)
- batch, submitting FTP requests in batch [88](#)
- BIG5
 - FTP subcommand [146](#)
 - LPR parameter [320](#)
- BINARY
 - FTP subcommand [147](#)
 - LPR parameter [320](#)
- blinking character attribute (GDDM/MVS) [334](#)
- BLOCK (FTP subcommand) [148](#)
- block mode (FTP) [220](#)
- BLocks (FTP LOCSITE and SITE parameter) [183](#), [252](#)
- BLOCKSIZE
 - FTP LOCSITE and SITE parameter [183](#), [252](#)
- BRK (TELNET subcommand) [11](#)
- BUFNO
 - FTP LOCSITE and SITE parameter [253](#)
- BURST (LPR parameter) [320](#)

C

- C (FTP MODE parameter) [220](#)
- carriage return, suppressing (TELNET) [15](#)
- CC
 - LPR parameter [320](#)
- CCC (FTP subcommand) [148](#)
- CCTrans (FTP.DATA parameter) [79](#)
- CD (FTP subcommand) [149](#)
- CDUP (FTP subcommand) [151](#)
- changing
 - directory of an MVS FTP server [149](#)

- changing (*continued*)
 - directory on a foreign host [149](#)
 - local site defaults using FTP.DATA [62](#)
 - to the parent of the current directory [151](#)
 - TSO user ID password [293](#)
 - working directory [175](#)
 - working level qualifier [175](#)
- character display (GDDM/MVS) [334](#)
- checkpointing
 - allow opening of data set [196](#)
 - prevent opening of data set [192](#)
 - restarting data transfer [242](#)
- CHKPTInt (FTP LOCSITE and SITE parameter) [253](#)
- class A network addresses [5](#)
- class, LPR parameter [320](#)
- CLEAR (FTP subcommand) [152](#)
- clearing the data path (TELNET) [14](#)
- client, description [2](#)
- CLOSE (FTP subcommand) [153](#)
- CMap (GDDMXD option) [338](#)
- codes
 - internal error [100](#)
 - reply [100](#)
 - return [96](#)
 - subcommand [97](#)
- color mixing (GDDM/MVS) [334](#)
- command (RSH parameter) [353](#)
- command_line (TSO parameter) [288](#)
- commands
 - FTP [4, 19](#)
 - GDDMXD [335](#)
 - PROFILE [32](#)
 - RExec [4](#)
 - RSH [352](#)
 - TELNET [3, 8](#)
- communication media [1](#)
- Communications Server for z/OS, commands [3](#)
- Communications Server for z/OS, online information [xxii](#)
- Compr (GDDMXD option) [339](#)
- COMPRESS (FTP subcommand) [153](#)
- compressed mode (FTP) [220](#)
- computer networks [1](#)
- connecting to a foreign host FTP server [30, 230](#)
- control characters, sending [14](#)
- COPIES (LPR parameter) [320](#)
- copying
 - data sets to a foreign host [238](#)
 - files from a foreign host [166](#)
 - multiple data sets to a foreign host [221](#)
 - multiple files from a foreign host [213](#)
- CPROTECT (FTP subcommand) [154](#)
- creating
 - a directory on a foreign host [216](#)
 - a PDS on the local host [177](#)
 - an input data set with the SQL query [131](#)
- CSSMTP
 - Creating mail messages on the JES spool data set [297](#)
 - Example of an undelivered mail notification [309](#)
 - Example of generated error reports [309](#)
 - Example of receiving mail [308](#)
 - exit [308](#)
 - SMTP commands
 - DATA command [304](#)
 - EHLO command [305](#)

- CSSMTP (*continued*)
 - SMTP commands (*continued*)
 - HELO command [305](#)
 - MAIL FROM command [306](#)
 - QUIT command [306](#)
 - RCPT TO command [307](#)
 - RSET command [307](#)
 - STARTTLS command [307](#)
 - Using the IEBGENER utility to copy a mail file to a JES sysout file [301](#)
 - Using the SMTPNOTE command [297](#)
 - Using the TSO TRANSMIT command to send a mail file [300](#)
- CYLINDERS (FTP LOCSITE and SITE parameter) [186, 255](#)

D

- data compression [220](#)
- data sets
 - FTP input [27](#)
 - FTP output [27](#)
 - NETRC.DATA [349, 354](#)
 - partitioned [359, 360](#)
 - sequential [45, 359, 360](#)
 - TSO [359](#)
- data sets transfer
 - using z/OS Communications Server [xv](#)
- data transfer
 - methods [45, 361](#)
 - types [45](#)
- data transfer type conversion
 - ASCII [146](#)
 - EBCDIC [164](#)
 - EUCKANJI [165](#)
 - HANGEUL [170](#)
 - IBMKANJI [171](#)
 - image [147](#)
 - JIS78KJ [172](#)
 - JIS83KJ [173](#)
 - KSC-5601 [173](#)
 - SJISKANJI [275](#)
 - TCHINESE [287](#)
- data_set (FTP LMKDIR parameter) [177](#)
- data_set_name (LPR parameter) [321](#)
- DATACLAS
 - FTP LOCSITE and SITE parameter [186, 255](#)
- DATASETMODE (FTP LOCSITE and SITE parameter) [36, 187, 256](#)
- Db2
 - FTP LOCSITE and SITE parameter [187, 256](#)
 - SQL queries with FTP [130](#)
 - subsystems in FTP [132](#)
- Db2 database [130](#)
- DBCS
 - DBCS support for FTP [78](#)
 - DBCS support for TELNET [16](#)
 - DBCS translation tables [79](#)
 - setting transfer type [79](#)
- DBCS subcommands
 - QUOTE [240](#)
 - TYPE [289](#)
 - TYPE aliases [80, 81](#)
- DCBDSN

DCBDSN (continued)

FTP LOCSITE and SITE parameter [187](#), [256](#)

DEBUG

FTP subcommand [154](#)

Telnet parameter [8](#)

Default Vector Symbol Set (GDDM/MVS) [334](#)

DELETE (FTP subcommand) [158](#)

deleting

a job [120](#)

files on a foreign host [158](#)

multiple files on a foreign host [211](#)

DELIMIT (FTP subcommand) [158](#)

DEST (FTP LOCSITE and SITE parameter) [258](#)

destination_file (FTP APPEND parameter) [144](#)

detached graphics cursor (GDDM/MVS) [334](#)

detectable fields (GDDM/MVS) [334](#)

determining a foreign host operating system [287](#)

DEV.NULL directory [151](#), [176](#)

devices, network [2](#)

DIR (FTP subcommand) [35](#), [36](#), [159](#), [161](#)

directories

changing the directory of an MVS FTP server [149](#)

changing the directory on the foreign host [149](#)

changing the working directory [175](#)

changing to the parent of the current directory [151](#)

creating a directory on a foreign host [216](#)

DEV.NULL [151](#), [176](#)

obtaining a list of directory entries [159](#)

removing a directory from a foreign host [244](#)

transferring PDS directory information [362](#)

working with directories on the foreign host [34](#)

working with directories on the local host [38](#), [39](#)

directory

FTP CD parameter [149](#)

FTP MKDIR parameter [216](#)

FTP RMDIR parameter [244](#)

DIRECTORY

FTP LOCSITE and SITE parameter [188](#), [259](#)

DIRECTORYMODE

FTP LOCSITE and SITE parameter [36](#), [188](#), [259](#)

disability [401](#)

disconnecting from a host using FTP [153](#)

DISK

FTP DIR parameter [159](#)

FTP LS parameter [208](#)

displaying

FTP help information [171](#)

local status information (FTP) [201](#)

TELNET help information [11](#)

the current working directory [239](#)

the current working level qualifier [208](#)

the file name delimiter [158](#)

the operating system name (FTP) [287](#)

the status of an FTP job [115](#)

DNS, online information [xxiii](#)

dotted decimal notation [2](#)

DUMP (FTP subcommand) [163](#)

E

E (FTP TYPE parameter) [289](#)

EBCDIC

FTP subcommand [164](#)

transferring binary data to ASCII [289](#)

EBCDIC (continued)

transferring binary data to EBCDIC [289](#)

transferring text data to ASCII [289](#)

transferring text data to EBCDIC [289](#)

electronic mail

e-mail [xv](#)

ending a TELNET session [13](#)

Enter (GDDMXD option) [339](#)

establishing

a connection to a foreign host [30](#), [31](#)

default working directory [32](#)

Ethernet protocol [1](#)

EUC (Extended UNIX Code) [165](#)

EUCKANJI

FTP subcommand [165](#)

LPR parameter [319](#)

examples

Generation Data Group (GDG) [85](#), [87](#)

JES [114](#)

LPQ [316](#)

LPR [326](#)

LPRM [329](#)

LPRSET [331](#)

SQL query output [134](#)

TELNET Help [12](#)

using TELNET to log on to a foreign host [9](#)

examples, FTP

APPEND [144](#)

differences between DIR and LS output [35](#), [36](#)

establishing a connection [31](#)

FTP as a batch job [90](#)

FTP EXEC [92](#), [93](#)

Get and MGet [46](#)

issuing subcommands from the EXEC interface [94](#)

LMKDIR [177](#)

MKDIR [216](#)

PUt and MPut [52](#)

showing the results of STATUS [278](#)

showing the results with and without DEBUG [154](#)

transferring data [45](#)

working with foreign directories [34](#)

working with local directories [39](#)

EXEC interface usage [92](#)

F

F

FTP STRUCT parameter [286](#)

FTP TYPE parameter [289](#)

FEATure (FTP subcommand) [166](#)

FILE (FTP subcommand) [166](#)

file name

delimiter [158](#)

obtaining a list [208](#)

specifying [359](#)

file transfer types

ASCII [45](#), [146](#), [289](#), [290](#), [361](#)

EBCDIC [45](#), [164](#), [289](#), [291](#), [361](#)

image [45](#), [147](#), [289](#), [292](#), [361](#)

kanji [45](#), [289](#), [291](#), [361](#)

files

AIX [362](#)

AS/400 [363](#)

specifying [359](#)

FILETYPE

FTP LOCSITE and SITE parameter [189](#), [261](#)

FILTER (LPR parameter) [321](#)

foreign_file

FTP DELETE parameter [158](#)

FTP GET parameter [166](#)

FTP MDELETE parameter [211](#)

FTP MGET parameter [213](#)

FTP PUT parameter [238](#)

foreign_host

FTP parameter [20](#)

RSH parameter [353](#)

TELNET parameter [8](#)

FTP

command [4](#), [19](#)

data conversion [77](#)

data transfer methods [45](#), [361](#)

Db2 subsystems for SQL queries [132](#)

DBCS support [78](#)

ddname support [56](#)

EXEC interface [92](#)

EXIT return codes [96](#)

FTP-supplied Db2 column headings [132](#)

internal error codes [100](#)

issuing subcommands from a data set [93](#)

logging on [24](#)

MBCS language support [81](#)

parameters [24](#)

reply codes [100](#)

restarting a failed data transfer [103](#)

return codes [95](#)

security considerations [39](#)

specifying data set attributes [82](#)

subcommand codes [97](#)

transferring data [19](#), [43](#), [45](#)

user-level options [62](#)

FTP Client, configuring for SOCKS server [63](#)

FTP examples

FTP as a batch job [90](#)

FTP EXEC [93](#)

Generation Data Group (GDG) [85](#), [87](#)

FTP format options

NOSPREAD [133](#)

SPREAD [133](#)

SQLCOL [131](#)

FTP requests in batch, JCL for [88](#)

FTP subcommands

! [143](#)

ACCT [144](#)

APPEND [144](#)

ASCII [146](#)

AUTH [146](#)

BIG5 [146](#)

BINARY [147](#)

BLOCK [148](#)

CCc [148](#)

CD [149](#)

CDUP [151](#)

CLEAR [152](#)

CLOSE [153](#)

COMPRESS [153](#)

CProtect [154](#)

DEBUG [154](#)

DELETE [158](#)

FTP subcommands (*continued*)

DELIMIT [158](#)

DIR [159](#)

DUMP [163](#)

EBCDIC [164](#)

EUCKANJI [165](#)

FEATure [166](#)

FILE [166](#)

GET [166](#)

GLOB [168](#)

HANGEUL [170](#)

HELP [171](#)

IBMKANJI [171](#)

JIS78KJ [172](#)

JIS83KJ [173](#)

KSC5601 [173](#)

LANGUAGE [174](#)

LCD [175](#)

LMKDIR [177](#)

LOCSITE [179](#)

LOCSTAT [201](#)

LPWD [208](#)

LS [208](#)

MDELETE [211](#)

MGET [213](#)

MKDIR [216](#)

MKFIFO [219](#)

MODE [220](#)

MPUT [221](#)

MVSGet [223](#)

MVSPut [227](#)

NOOP [230](#)

OPEN [230](#)

PASS [231](#)

PRIVATE [233](#)

PROMPT [234](#)

PROTECT [234](#)

PROXY [235](#)

PUT [238](#)

PWD [239](#)

QUIT [240](#)

QUOTE [240](#)

RECORD [241](#)

RENAME [242](#)

RESTART [242](#)

RMDIR [244](#)

SAfe [244](#)

SCHINESE [244](#)

SENDPORT [245](#)

SENDSITE [246](#)

SITE [247](#)

SJISKANJI [275](#)

SRESTART [276](#)

STATUS [278](#)

STREAM [285](#)

STRUCTURE [286](#)

SUNIQUE [286](#)

SYSTEM [287](#)

TCHINESE [287](#)

TSO [288](#)

TYPE [289](#)

UCS2 [293](#)

USER [293](#)

Verbose [296](#)

FTP-supported SQL data types

- CHAR [130](#)
- DATE [130](#)
- DECIMAL [130](#)
- FLOAT [130](#)
- INTEGER [130](#)
- LONG VARCHAR [130](#)
- SMALLINT [130](#)
- TIME [130](#)
- TIMESTAMP [130](#)
- VARCHAR [130](#)

FTP.DATA data set [62, 63](#)

function keys for TELNET [15](#)

G

GColornn (GDDMXD option) [339](#)

GDDM

- application limitations [333](#)
- displaying graphics on X Windows workstations [xv](#)
- GDDM display limitations [334](#)

GDDMXD

- CLIST [333, 335](#)
- Graphics Window [336](#)

GDDMXD/MVS

- APL2 character set keyboard [345, 365](#)
- GDXAPLCS.MAP [345](#)
- keyboard functions [344](#)
- overview [333](#)
- target display, identifying [335](#)
- TSO EXEC command [335](#)
- usage [336](#)
- using [333, 335](#)
- X.DEFAULTS data set [336](#)

GDDMXD/MVS user-specified options

- ANFontn [337](#)
- CMap [338](#)
- Compr [339](#)
- GColornn [339](#)
- Geometry [341](#)
- GMCPnn [341](#)
- HostRast [342](#)
- XSync [343](#)
- ZWL [343](#)

GDDMXD/MVS with X Windows [333](#)

Generation Data [85](#)

Generation Data Group Support (GDG) [85](#)

Geometry (GDDMXD option) [341](#)

Get FTP subcommand [46](#)

GET FTP subcommand [166](#)

getting started [1](#)

GLOB (FTP subcommand) [168](#)

GMCPnn (GDDMXD option) [341](#)

H

HANGEUL

- FTP subcommand [170](#)
- LPR parameter [322](#)

HEADER, LPR parameter [322](#)

host

- foreign [34](#)
- local [38](#)

host (*continued*)

- LPQ parameter [316](#)
- LPR parameter [322](#)
- LPRM parameter [329](#)
- names [5](#)
- remote [2, 7, 34, 149, 153, 216, 221, 223, 227, 238, 329, 330, 363](#)

host_name (FTP OPEN parameter) [231](#)

HostRast (GDDMXD option) [342](#)

hosts, using other [4](#)

how TCP/IP uses networks [2](#)

I

I (FTP TYPE parameter) [289](#)

IBM 3179G device model [334](#)

IBM Software Support Center, contacting [xvi](#)

IBMKANJI (FTP subcommand) [171](#)

identifying

- the target display (GDDM/MVS) [335](#)
- yourself to a host [293](#)

INDENT (LPR parameter) [322](#)

Information APARs [xx](#)

interfaces, EXEC [92](#)

interfacing with JES [114](#)

internal error codes, FTP [100–102](#)

internet_address (XWINDOWS DISPLAY parameter) [335](#)

Internet, finding z/OS information online [xxii](#)

interrupting, the current process (TELNET) [12](#)

IP (TELNET subcommand) [12](#)

IPv4/IPv6 addressing [2](#)

ISPFStats

- LOCSite [190](#)

- PDS member [85](#)

issuing FTP subcommands from a data set [93](#)

issuing FTP subcommands from the EXEC interface [94](#)

J

JCL [88, 114](#)

JCL for submitting FTP requests in batch [88](#)

JES

- deleting a job [120](#)

- displaying job status [115](#)

- interfacing with [114](#)

- receiving spool output [118](#)

- submitting a job [114](#)

- terminating access to [121](#)

JESLRECL (SITE parameter) [261](#)

JESRECFM (SITE parameter) [262](#)

JIS78KJ (FTP subcommand) [172](#)

JIS83KJ (FTP subcommand) [173](#)

JISROMAN

- FTP JIS78KJ parameter [172](#)

- FTP JIS83KJ parameter [173, 174](#)

JNum (LPR parameter) [322](#)

JOB (LPR parameter) [322](#)

Job Scheduler [114](#)

JOB_ID

- LPQ parameter [316](#)

- LPRM parameter [329](#)

K

kanji

- EUCKANJI [165](#)
- IBMKANJI [171](#)
- SJISKANJI [275](#)

keyboard [401](#)

KSC5601

- FTP subcommand [173](#)
- LPR parameter [323](#)

L

LAN (local area network) [1](#)

LANDSCAPE (LPR parameter) [323](#)

LANGUAGE (FTP subcommand) [174](#)

LCD (FTP subcommand) [175](#)

leaving the FTP environment [240](#)

license, patent, and copyright information [405](#)

line feed, suppressing (TELNET) [15](#)

Line Printer

- Client (LPR) [315](#)

- Daemon (LPD) [315](#)

LINECOUNT (LPR parameter) [323](#)

Linemode (TELNET parameter) [8](#), [15](#)

LMKDIR (FTP subcommand) [177](#)

LMTR (load module transfer) [59](#)

load module transfer (LMTR) [59](#)

local host [38](#), [177](#), [179](#), [288](#)

local node, description [2](#)

local_data_set

- FTP APPEND parameter [144](#)

- FTP MPUT parameter [221](#)

- FTP PUT parameter [238](#)

local_file (FTP GET parameter) [166](#)

LOCSITE (FTP subcommand) [179](#)

LOCSITE parameters [179](#)

LOCSTAT (FTP subcommand) [201](#)

logging on

- to a host using Telnet [7](#)

- to FTP [24](#)

- to other hosts [3](#)

LOOPBACK [5](#)

LPD (line printer daemon) [315](#)

LPQ (remote printing command)

- description [5](#), [315](#)

- examples [316](#)

- usage [317](#)

LPR (remote printing command)

- description [5](#), [317](#)

- examples [326](#)

- usage [327](#)

LPRM (remote printing command)

- description [5](#), [328](#)

- examples [329](#)

- usage [330](#)

LPRSET (remote printing command)

- description [5](#), [330](#), [331](#)

- examples [331](#)

- usage [331](#)

LPWD (FTP subcommand) [208](#)

LRECL

- FTP LOCSITE and SITE parameter [190](#), [262](#), [263](#)

ls (FTP subcommand) [35](#), [36](#), [208](#)

M

MAIL (LPR parameter) [324](#)

mainframe

- education xx

mapping values [345](#), [365](#)

MBCS translation tables, and FTP [82](#)

MDELETE (FTP subcommand) [211](#)

MGet (FTP subcommand) [46](#)

MGET (FTP subcommand) [213](#)

MGMTCLAS

- FTP LOCSITE and SITE parameter [191](#), [264](#)

MIGRATEVOL

- FTP LOCSITE and SITE parameter [191](#), [264](#)

MKDIR (FTP subcommand) [216](#)

MKFifo (FTP subcommand) [219](#)

MODE (FTP subcommand) [220](#)

MPut (FTP subcommand) [52](#)

MPUT (FTP subcommand) [221](#)

N

NAME

- FTP DIR parameter [159](#)

- FTP LS parameter [208](#)

- FTP STATUS parameter [278](#)

- LPR parameter [324](#)

name server, description [3](#)

named pipes, MKFifo subcommand [219](#)

names

- host [5](#)

- network, description [3](#)

- printer [5](#)

NETRC.DATA data set [349](#)

network address format [2](#)

network devices [2](#)

network names [3](#)

network protocols

- 802.3 [1](#)

- Ethernet [1](#)

- SNA [1](#)

- token ring [1](#)

- X.25 [1](#)

networks

- TCP/IP [2](#)

new_name (FTP RENAME parameter) [242](#)

NewLiner (GDDMXD option) [342](#)

NOASAtans parameter [264](#)

NOAUTOMOUNT (FTP LOCSITE and SITE parameter) [191](#), [264](#)

NOAUTORECALL (FTP LOCSITE and SITE parameter) [191](#), [264](#)

NOBINARY (LPR parameter) [324](#)

NOBURST (LPR parameter) [320](#)

NOCC

- LPR parameter [320](#)

nodes, descriptions [1](#), [2](#)

noheader, LPR parameter [320](#)

NOLinecount (LPR parameter) [322](#)

NOOP

- FTP subcommand [230](#)

NOPOSTSCRIPT (LPR parameter) [324](#)

NORDW (FTP LOCSITE parameter) [192](#)

NORESTGet (FTP LOCSITE parameter) [192](#)

- NOSPREAD
 - FTP format option [133](#)
 - FTP LOC SITE and SITE parameter [193](#), [265](#)
- notation system, dotted decimal [2](#)
- NOTOPMARGIN (LPR parameter) [324](#)
- NOTRAILINGBLANKS (FTP SITE parameter) [265](#)
- NOTYPE
 - FTP EUCKANJI parameter [165](#)
 - FTP HANGEUL parameter [170](#)
 - FTP IBMKANJI parameter [171](#)
 - FTP JIS78KJ parameter [172](#)
 - FTP JIS83KJ parameter [173](#), [174](#)
 - FTP KSC5601 parameter [174](#)
 - FTP SJISKANJI parameter [275](#)
 - FTP TCHINESE parameter [287](#)
- NOWRAPRECORD (FTP LOC SITE and SITE parameter) [193](#), [266](#)

O

- obtaining
 - a list of directory names [159](#)
 - a list of file names [208](#)
 - status and system information [33](#)
- OFF
 - GDDMXD parameter [335](#)
- ON
 - GDDMXD parameter [335](#)
- OPEN (FTP subcommand) [230](#)
- original_name (FTP RENAME parameter) [242](#)
- overviews
 - differences between DIR and LS output [35](#)
 - GDDMXD/MVS [333](#)

P

- PA1 (TELNET subcommand) [13](#)
- parameter
 - FTP LOC SITE parameter [179](#)
 - FTP SITE parameter [247](#)
- parameters, FTP
 - EXIT [20](#)
 - FOREIGN_HOST [20](#)
 - PORT_NUMBER [20](#)
 - TCP [20](#)
 - TIMEOUT [20](#)
 - TRACE [20](#)
 - TRANSLATE [20](#)
- parameters, SNALINK LU6.2 [231](#)
- passing TSO commands to your local host [288](#)
- password
 - FTP PASS parameter [231](#)
 - FTP USER parameter [294](#)
- password, use with FTP
 - ACCT [144](#)
 - PASS [231](#)
 - USER [294](#)
- PDS [177](#), [359](#), [360](#), [362](#)
- performing a Db2 SQL query
 - from an FTP client [133](#)
 - from an FTP server [134](#)
 - with FTP [130](#)
- physical network, description [1](#)

- pixel spacing (GDDM/MVS) [334](#)
- port numbers, description [3](#)
- port_number
 - FTP OPEN parameter [231](#)
 - FTP parameter [20](#)
 - TELNET parameter [8](#)
- ports, description [3](#)
- POSTSCRIPT (LPR parameter) [325](#)
- preparing the FTP environment [43](#)
- prerequisite information [xx](#)
- primary
 - FTP LOC SITE and SITE parameter [194](#), [266](#)
- printer
 - LPQ parameter [316](#)
 - LPR parameter [324](#)
 - LPRM parameter [329](#)
 - names [5](#)
- printer_host (LPRSET parameter) [330](#)
- printing
 - remote xv, [315](#)
 - to or from other hosts [4](#)
- PRIVATE (FTP subcommand) [233](#)
- PROFILE command [32](#)
- PROMPT (FTP subcommand) [234](#)
- PROTECT (FTP subcommand) [234](#)
- protocols
 - description [1](#)
 - File Transfer Protocol [19](#), [43](#)
 - Telnet Protocol [7](#)
 - X Window System Protocol [333](#)
 - z/OS UNIX Remote Execution Protocol [347](#), [355](#)
- PROXY
 - bounce attack [237](#)
 - FTP subcommand [235](#)
 - security [237](#)
- PUT (FTP subcommand) [223](#), [227](#), [238](#)
- PUTt (FTP subcommand) [52](#)
- PWD (FTP subcommand) [32](#), [239](#)

Q

- Qdisk (FTP LOC SITE and SITE parameter) [194](#), [267](#)
- qualifier (FTP LCD parameter) [175](#)
- query, LPRSET parameter [330](#)
- querying a connection (TELNET) [11](#)
- QUIT
 - FTP subcommand [240](#)
 - TELNET subcommand [13](#)
- QUOTE
 - DBCS subcommand [240](#)
 - FTP subcommand [240](#)
- QUOTESoverride, FTP LOC SITE parameter [194](#)

R

- RACF [5](#), [32](#)
- RDW
 - FTP LOC SITE parameter [194](#)
- READTAPEFormat, FTP LOC SITE parameter [194](#)
- READTAPEFormat, FTP Site parameter [267](#)
- receiving spool output
 - in a group [119](#)

- receiving spool output (*continued*)
 - individually [118](#)
- RECFM
 - FTP LOCSITE and SITE parameter [179](#), [268](#)
- RECORD (FTP subcommand) [241](#)
- Record Descriptor Words (RDW) [192](#)
- remote host logon [xv](#)
- remote node, description [2](#)
- remote printing [315](#)
- remote printing commands
 - LPQ [315](#)
 - LPR [317](#)
 - LPRM [328](#)
 - LPRSET [330](#), [331](#)
- removing a directory (FTP) [244](#)
- RENAME (FTP subcommand) [242](#)
- renaming files on a foreign host [242](#)
- REPLACE
 - FTP GET parameter [166](#)
 - FTP MGET parameter [213](#)
- requests in batch, submitting FTP [88](#)
- resizing the GDDMXD graphics window [336](#)
- RESTART (FTP subcommand) [242](#)
- restarting a checkpointed data transfer [242](#)
- RESTGet (FTP LOCSITE parameter) [196](#)
- RETPD
 - FTP LOCSITE and SITE parameter [196](#), [268](#)
- retrieving status information from a remote host [278](#)
- return codes, FTP [96](#)
- REXEC [347](#), [355](#)
- REXX command list language [24](#)
- RFC (request for comments)
 - accessing online [xxii](#)
- RHOSTS.DATA data set [354](#)
- RMDIR (FTP subcommand) [244](#)
- RSH [352](#)

S

- S (FTP MODE parameter) [220](#)
- SAFE (FTP subcommand) [244](#)
- sample FTP.DATA data set [63](#)
- SBCS translation tables, and FTP [77](#)
- SBTRANS (FTP.DATA parameter) [79](#)
- SCHINESE
 - FTP subcommand [244](#)
 - LPR parameter [324](#)
- SECONDARY
 - FTP LOCSITE and SITE parameter [198](#), [270](#)
- sending
 - ASCII control characters to a host in line mode [14](#)
 - break or attention keystroke to a host [11](#)
 - data using the QUOTE subcommand [240](#)
 - electronic mail using SMTP commands [4](#)
 - PA1 keystroke to a host [13](#)
 - site-specific information to a host [247](#)
 - uninterpreted string of data [240](#)
- sendmail to CSSMTP bridge [311](#)
- SENDPORT (FTP subcommand) [245](#)
- SENDSITE (FTP subcommand) [246](#)
- sequential data sets [360](#)
- server
 - description [2](#)
 - FTP HELP parameter [171](#)

- setting
 - characteristics for an SQL query [131](#)
 - data set or file structure [286](#)
 - data transfer mode [220](#)
 - data transfer type [79](#), [289](#)
- setting up tcpipv3r1.GDXAPLCS.MAP [345](#)
- shortcut keys [401](#)
- SITE (FTP subcommand) [247](#)
- SITE parameters [247](#)
- SJISKANJI
 - FTP subcommand [275](#)
 - LPR parameter [324](#)
- SMS [82](#), [183](#), [252](#)
- SMTP
 - description [4](#)
 - mail forwarding [297](#)
- SNA network protocol [1](#)
- softcopy information [xx](#)
- specifying
 - column headings for an SQL query [132](#)
 - data sets and files [359](#)
 - report format of your output data set [133](#)
 - site information to the local host [179](#)
 - spreadsheet format of your output data set [133](#)
 - the Db2 subsystem to perform a query [132](#)
 - values for new data sets [82](#)
- SPREAD
 - FTP LOCSITE and SITE parameter [133](#), [198](#), [270](#)
- SQL
 - FTP-supported data types [130](#)
 - imbedded statements [130](#)
 - with FTP on the client [133](#)
 - with FTP on the server [134](#)
- SQL data type [130](#)
- SQLCOL
 - FTP LOCSITE and SITE parameter [198](#), [270](#)
- SRestart (FTP subcommand) [276](#)
- STATUS (FTP subcommand) [278](#)
- status and system information [33](#)
- STORCLASS
 - FTP LOCSITE and SITE parameter [198](#), [271](#)
- store command (STOR) [286](#)
- STREAM (FTP subcommand) [285](#)
- stream mode (FTP) [220](#)
- string (FTP QUOTE parameter) [240](#)
- STRUCT (FTP subcommand) [286](#)
- SUBCOMMAND (FTP HELP parameter) [171](#)
- submitting
 - FTP requests in batch [88](#)
 - job and automatically receiving output [120](#)
 - job using FTP [114](#)
 - requests without input and output data sets [91](#)
 - REXEC requests in batch [350](#)
 - SQL query using FTP [133](#)
- summary of changes [xxv](#)
- SUNIQUE (FTP subcommand) [286](#)
- supplying
 - a password to a foreign host [231](#)
 - account information to a foreign host [144](#)
- suppressing carriage return and line feed [15](#)
- SYNCH (TELNET subcommand) [14](#)
- syntax diagram, how to read [xvii](#)
- SYSTEM (FTP subcommand) [287](#)

T

- tables
 - translation [79](#)
- target display (GDDM/MVS) [335](#)
- target_screen (XWINDOWS DISPLAY parameter) [335](#)
- target_server (XWINDOWS DISPLAY parameter) [335](#)
- tasks
 - GDDMXD/MVS
 - overview [333](#)
 - submitting a job
 - steps for [114](#)
 - submitting a job and automatically receiving output
 - steps for [120](#)
 - Using a DCBDSN model to create a new data set
 - steps for [84](#)
- TCHINESE
 - FTP subcommand [287](#)
 - LPR parameter [324](#)
- TCP, FTP parameter [20](#)
- TCP/IP
 - addresses [2](#)
 - commands [3](#)
 - description [1](#)
 - layers [2](#)
 - networks [2](#)
 - online information [xxii](#)
 - protocol specifications [381](#)
 - understanding [1](#)
- Technotes [xx](#)
- TELNET
 - 3270 DBCS Transform Mode [16](#)
 - command [3](#), [8](#)
 - supported display stations [9](#)
- TELNET examples
 - command format [8](#)
 - logging on to a foreign host [9](#)
 - logging onto a host using [7](#)
 - using Help [12](#)
- TELNET function keys
 - in line mode [15](#)
 - in transparent mode [15](#)
- TELNET parameters
 - DEBUG [8](#)
 - foreign_host [8](#)
 - Help [8](#)
 - Linemode [8](#)
 - port_number [8](#)
 - TRANslate data_set_name [8](#)
- Telnet Protocol [7](#)
- TELNET subcommands
 - AO [10](#)
 - AYT [11](#)
 - BRK [11](#)
 - HELP [11](#)
 - IP [12](#)
 - PA1 [13](#)
 - QUIT [13](#)
 - SYNCH [14](#)
- terminating
 - access to JES [121](#)
 - output of TELNET information [10](#)
- testing
 - commands with loopback [5](#)

- testing (*continued*)
 - FTP connection [230](#)
 - throughput with *DEV.NULL [151](#), [176](#)
- Timeout (FTP parameter) [20](#)
- TITLE (LPR parameter) [325](#)
- toggling
 - internal debug options (FTP) [154](#)
 - sending of port information [245](#)
 - sending of site information [246](#)
 - storage method [286](#)
- token ring network protocol [1](#)
- TOPMARGIN (LPR parameter) [325](#)
- TRACE
 - FTP parameter [20](#)
 - LPQ parameter [316](#)
 - LPR parameter [325](#)
 - LPRM parameter [329](#)
 - LPRSET parameter [330](#)
- TRACKS (FTP LOCSITE and SITE parameter) [199](#), [271](#)
- trademark information [408](#)
- TRAILINGBLANKS, FTP SITE parameter [271](#)
- transferring
 - data sets between hosts [4](#)
 - data using FTP [19](#), [43](#), [45](#)
 - DBCS data sets with FTP [79](#)
 - PDS directory information [362](#)
- TRANslate data_set_name
 - FTP parameter [20](#)
 - TELNET parameter [8](#)
- TRANSLATETABLE (LPR parameter) [325](#)
- transparent mode [15](#)
- TSO
 - entering TCP/IP commands [1](#)
 - FTP subcommand [288](#)
 - Session Manager [336](#)
- TSO commands
 - PROFILE [32](#)
- TYPE
 - DBCS subcommand [289](#)
 - FTP subcommand [289](#)
 - LPQ parameter [316](#)
 - LPR parameter [325](#)
 - LPRM parameter [329](#)
 - LPRSET parameter [330](#)

U

- UCS2 (FTP subcommand) [293](#)
- UNIT (FTP LOCSITE and SITE parameter) [200](#), [273](#)
- Unit of Work (UOW) [4](#)
- USCFXLATE (LPR parameter) [326](#)
- user
 - ID [5](#)
 - password [5](#)
- USER (FTP subcommand) [293](#)
- USER name (FTP parameter) [325](#)
- user_id (FTP USER parameter) [294](#)
- user-specified options (GDDM/MVS) [336](#)
- uses of TCP/IP
 - data transfer [4](#), [19](#), [43](#)
 - electronic mail [4](#)
 - printing on other hosts [4](#), [315](#)
 - remote login [3](#), [7](#)
 - using other hosts [4](#), [347](#), [355](#)

V

VERBOSE (FTP subcommand) [296](#)
VERSION
 LPQ parameter [316](#)
 LPR parameter [326](#)
 LPRM parameter [329](#)
 LPRSET parameter [330](#)
visual appearance (GDDM/MVS) [334](#)
VM files [363](#)
VOLUME
 FTP LOCSITE and SITE parameter [201](#), [273](#)
VTAM [13](#)
VTAM, online information [xxii](#)

W

WAN (wide area network) [1](#)
well-known ports, description [3](#)
what you need to get started [5](#)
WIDTH (LPR parameter) [326](#)
working directory [32](#), [175](#)
working with directories
 on the foreign host [34](#)
 on the local host [38](#), [39](#)
working-level qualifier [175](#), [208](#)
WRAPRECORD
 FTP LOCSITE and SITE parameter [201](#), [274](#)

X

X Window System [333](#)
X.25 network protocol [1](#)
XLATETABLE (LPR parameter) [326](#)
XSync (GDDMXD option) [343](#)

Z

z/OS Basic Skills Information Center [xx](#)
z/OS UNIX orexec/rexec [355](#)
z/OS UNIX orsh/rsh [357](#)
z/OS UNIX REXEC
 command [4](#), [347](#), [355](#)
 format [347](#), [355](#)
 requests, submitting in batch [350](#)
z/OS UNIX RSH
 command [4](#), [347](#), [355](#)
 format [357](#)
 requests, submitting in batch [350](#)
z/OS, documentation library listing [409](#)
ZWL (GDDMXD option) [343](#)

Communicating your comments to IBM

Important: If your comment regards a technical question or problem, see instead [“If you have a technical problem”](#) on page 423.

Submit your feedback by using the appropriate method for your type of comment or question:

Feedback on z/OS function

If your comment or question is about z/OS itself, submit a request through the [IBM RFE Community](#) (www.ibm.com/developerworks/rfe/).

Feedback on IBM Documentation function

If your comment or question is about the IBM Documentation functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Documentation Support at ibmdocs@us.ibm.com.

Feedback on the z/OS product documentation and content

If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to mhvrcfs@us.ibm.com. We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The title and order name of the document, and the version of z/OS Communications Server
- The section title of the specific information to which your comment relates
- The text of your comment.

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

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

If you have a technical problem

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the [IBM Support Portal](http://support.ibm.com) (support.ibm.com).
- Contact your IBM service representative.
- Call IBM technical support.



Product Number: 5650-ZOS

SC27-3662-50

