Client FTP2

This chapter describes Client FTP2, the File Transfer Protocol (FTP) that allows file transfers among unlike hosts in diverse internetworking environments. It contains these sections:

- About File Transfer Protocol

  Provides a brief overview of the Cisco IOS for S/390 File Transfer Protocol implementation.

- Client FTP2

  Provides a flow chart of how FTP works.

- Invoking Client FTP2

  Describes how to use Client FTP2 as both a TSO command processor and as a regular batch program.

- Client FTP2 Invocation Options

  Describes both the general and debug options available with the FTP2 command.

- General Client FTP2 Operation

  Describes the general operation of the Client FTP2 program.

- Client FTP2 Commands

  Describes each of the Client FTP2 commands and includes a table listing all the commands with a brief description of each.

Commands are:

? Command	! Command	$ Command	ABOR	ALLO
APPEND	ASCII	BINARY	BYE	CD
CDUP	CLOSE	DEBUG	DELETE	DIR
DISCONNECT	DO	EBCDIC	END	EXPE
FIREWALL	GET	HELP	LOG	LQUOTE
LS	MACDEF	MKDIR	MODE	NTRANS
OPEN	PUT	PWD	QUIT	QUOTE
RECV	REMHELP	REMSITE	REMSNDS	REMSTAT
RENAME	REST	RMDIR	SEND	SITE
SNDS	STATUS	STRUCT	SUNIQUE	TYPE

- Restart Support

  Describes how to restart an interrupted file transfer.

- Client FTP2 File Transfer Examples

  Provides examples of some of the Client FTP2 commands.

About File Transfer Protocol

The File Transfer Protocol (FTP) is an application protocol in the Internet protocol suite. It allows file transfers among unlike hosts in diverse internetworking environments. Using FTP, you can move a file from one computer to another, even if each computer has a different operating system and file storage format. Files can contain data, programs, text, or anything that can be stored online.

The objectives of the FTP protocol are to

- provide sharing of files (computer programs and/or data)
- encourage indirect or implicit use of remote computers through programs
- shield users from variations in file storage systems among hosts
- transfer data reliably and efficiently

FTP is based on a model of files having a few attributes and a mechanism for processing commands and replies. The command-and-reply mechanism establishes the parameters for a file transfer, then performs the transfer. Like Telnet, FTP runs over TCP and assumes the service level provided by TCP.

These documents define FTP:

- *RFC 959, File Transfer Protocol (FTP)*
- *MIL-STD-1780, Military Standard File Transfer Protocol*

Client FTP/Client FTP2

Two versions of Client FTP are provided with Cisco IOS for S/390: Client FTP2 and Client FTP.

Client FTP2 operates as a three-party model in which there are two control connections and one data connection. The FTP2 control connection and the local host connection are one and the same.

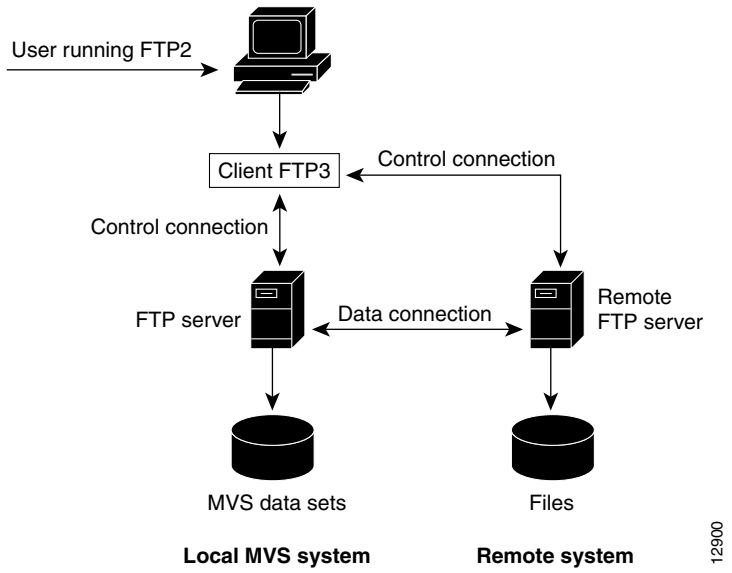
The FTP protocols require that you connect to a host before most commands can be issued. With the Client FTP2 program, an MVS TSO or batch user signs on to the remote host only. Client FTP2 automatically connects and logs on to the Cisco IOS for S/390 MVS host as the local host (thus hiding one side of the three-party model). So after a single sign-on to the remote host, the user can transfer files between the local host and the remote host.

Although Client FTP and Client FTP2 both operate internally as third-party models, Client FTP2 appears to you, the user, to be a two-party model. Also, Client FTP2 commands are designed to resemble UNIX FTP commands; therefore, Client FTP and Client FTP2 commands differ somewhat.

Client FTP2

Figure 2-1 shows the relationship between the Client FTP2 program and two FTP servers in the *three-party model*, indicating that three connections are maintained. Two connections are for command control and one is for data flow.

Figure 2-1 Client FTP2 program in the Three-Party Model



In Figure 2-1, the user is logged in to the local MVS system and is running FTP2 under TSO. Through FTP2, the user opens a connection to the remote FTP server. The connection to the local FTP server is opened automatically by the Client FTP2 program and is hidden from the user. FTP2 maintains two control connections: One for communication with the Cisco IOS for S/390 FTP server and the second with the remote FTP server.

With FTP2, the user has access to files on both systems. Data transfer occurs directly between the local and remote servers. Data does not pass through the Client FTP2 application.

Invoking Client FTP2

The Client FTP2 program runs as both a TSO command processor and as a regular batch program. This means that you can use Client FTP2 as a TSO command or call it as a regular program with MVS JCL.

Invoking Client FTP2 through TSO

In a TSO environment, Client FTP2 can be accessed as a TSO command processor or can be called as a program with the TSO **CALL** command. Because Client FTP2 does not use full-screen facilities, it can be used from any type of terminal supported by TSO, including 3270 systems, 3767 systems, and asynchronous ASCII terminals supported by NTO or NPSI.

Note You must have PROMPT set in your TSO profile for FTP2 to work properly in interactive mode.

FTP2 Command Processor

Invoke Client FTP2 by entering the **FTP2** TSO command in this format.

FTP2 [*/ option1 option2 ..*]

Syntax Description

ftp2 Invokes the Client FTP2 program.

/options Any number of FTP2 invocation options can be specified in the invocation command.

Client FTP2 responds with either of these messages:

```
Client FTP2 Enter command or '?'
```

or

```
Setting local directory to the TSO profile prefix
250 "USER1." is current prefix
Client FTP2 Enter command or '?'
```

At this point, you are automatically connected and logged onto the MVS host. The working directory on the local host is set to the TSO profile prefix. If no prefix is defined, the working directory is set to the TSO user ID.

FTP2 invocation options are described in Client FTP2 Invocation Options.

Usage Guidelines

All Client FTP2 options must be preceded by a slash (/). If the slash is omitted, the options are not recognized by Client FTP2.

TSO CALL Command

Use the tso **CALL** command in a TSO environment to call and execute the FTP2 program out of a specific library. This is especially useful at sites that run multiple releases of the product or have test and production versions of the product at different maintenance levels.

CALL 't01tcp.link(ftp2)' ['/option1 option2 ..']

Syntax Description

't01tcp.link(ftp2)'	Version of FTP2 being used.
'/options'	Any number of FTP2 invocation options can be included in the **CALL** command.

Usage Guidelines

The data set name, t01tcp.link, might need to be replaced by the appropriate data set name at your installation. Check with your Cisco IOS for S/390 site administrator.

When options are specified in the command statement, they must be enclosed in single quotes.

When invoked by the **CALL** command, Client FTP2 runs as a program and not as a TSO command processor. It is not necessary to allocate data sets before calling FTP2 with the **CALL** command.

Under TSO, any data set needed by the Client FTP2 program is dynamically allocated and freed.

See Client FTP2 Invocation Options for detailed descriptions of the FTP2 invocation options.

Batch Invocation

The Client FTP2 program can be run in batch as either a program like any other, or as a TSO command processor by running it under a batch Terminal Monitor Program (TMP). FTP2 commands **DO**, **DIR**, and **LS** require a TSO environment to execute, thus they only execute in batch under the TMP.

Note When running in batch mode, specify the Client FTP2 commands carefully, because the slightest error can cause all subsequent commands to fail and force you to rerun the batch job. For PL/I V2.2.1 users, FMID PLIX150 must be installed. The Cisco IOS for S/390 LINK library must be placed before PL/I runtime libraries in your STEPLIB concatenation for client batch jobs (including FTP, FTP2, ACPEEP, or TELNET). Failure to do this may cause IBM002I, IBM004I, or IBM014I messages.

When run in batch, Client FTP2 sets a program condition code depending on the severity of Client FTP2 errors encountered; a return code of 0 indicates that all commands entered were successful.

You can specify a NETRC file in batch mode. Specify a NETRC DD file with the name of your NETRC file. Use the NETRC invocation option described in Client FTP2 Invocation Options.

Batch Program

You can invoke Client FTP2 in batch in a manner similar to any other batch utility program. Invoking FTP2 as a batch program does not provide a TSO environment that the **DO**, **DIR**, and **LS** commands need. Thus these commands are rejected when FTP2 executes as a batch program.

```
//jobname    JOB job_stmt_parms
//FTPSTEP    EXEC PGM=FTP2,PARM='/ option1 option2...'
//STEPLIB    DD DSN=T01TCP.LINK,DISP=SHR
//SYSPRINT   DD SYSOUT=*
//SYSPUT     DD SYSOUT=*,DCB=BLKSIZE=133
//SYSGET     DD *,DCB=BLKSIZE=80
open unix
user pass
mkdir /u/lpn/d.new
```

Optionally, you can include a NETRC file, as shown here:

```
//NETRC DD DISP=SHR,DSN=userid.FTP.NETRC
```

Read Client FTP2 General Options for detailed descriptions of the FTP2 options.

Batch TMP

You can invoke Client FTP2 in batch as a TSO command processor by running it under a batch TMP.

In the following example, the batch TMP program is IKJEFT01, which is the normal TSO TMP. Since the batch TMP provides the TSO environment, all FTP2 commands are available.

For this example, the user programs are made available through the link list.

```
//jobname     JOB job_stmt_parms
//* JOB TO RUN FTP IN BATCH
//EXEC PGM=IKJEFT01,REGION=3000K
//SYSTSPRT   DD SYSOUT=X
//SYSPRINT   DD SYSOUT=X
//SYSPUT     DD *,DCB=BLKSIZE=80
open unix
user pass
mkdir /u/lpn/d.new
//SYSTSIN DD *
FTP2 / option1 option2...
```

Client FTP2 Invocation Options

This section describes the Client FTP2 invocation options and their requirements. These general notes apply to the Client FTP2 invocation options:

- No invocation options are required for the Client FTP2 program. If you specify any, they must be preceded by a slash (/) to meet the conventions of the PL/I runtime support package.
- An option name can immediately follow a slash, as in /FIOS.
- Invocation options are not case sensitive.
- Specify option names exactly as shown. Abbreviations are not permitted.

Client FTP2 General Options

This section describes the Client FTP2 general invocation options.

APP

The APP option (in the form APP=*vtam_application_name*) identifies the VTAM application name where the client connects. *vtam_application_name* is a 1- to 8-character name.

If the APP option is used, the SYS invocation option is ignored.

FIOS

Normally the FTP program interacts with the user through a terminal. The FIOS option lets the program read and execute a file containing commands and sends the results to a different file. Commands are read from a sequential file allocated to the SYSGET DD statement; execution results are written to a sequential file allocated to the SYSPUT DD statement. Under TSO, the files can be allocated as shown here:

ALLOCATE FILE(SYSGET) DATASET(*input_dataset*)

ALLOCATE FILE(SYSPUT) DATASET(*output_dataset*)

The input file should

- include all the necessary commands (such as **OPEN** and **LOG**)
- be unnumbered

The output file should have a blocksize of 133.

HOST

The HOST option identifies the remote host with which you want to connect. If you include the remote host name specified here in the NETRC file (userid.FTP.NETRC), the program takes the user ID and password from the NETRC file. Otherwise, you are prompted for a user ID and password.

HOST=*hostname*

Syntax Description

hostname A variable that specifies the name of the host.

LOGT

The LOGT option displays the current time before each line is sent to the terminal. This option is automatically set if either the test or FIOS option is specified.

NETRC

The NETRC option indicates a NETRC file is to be used to resolve the remote host ID and password. For batch TMP and straight batch FTP2, the netrc parameter must be coded if a NETRC file is to be used. Additionally, straight batch FTP2 requires a //NETRC DD statement in the job stream pointing to the NETRC file. Interactive and batch TMP users can also pre-allocate the NETRC file by coding an ALLOC statement with NETRC as the DD name. This is useful if the default naming conventions for the NETRC file are not being used.

NOA

No Automatic Logon (noa) specifies that there is no automatic connection to the local host (Cisco IOS for S/390 on MVS).

NOFIRE

The NOFIRE (No Firewall) option indicates that client FTP2 should disregard the Firewall-Friendly FTP RFC1579. The default is to implement FRC1579.

Additionally, the **FTP2 FIREWALL** command may be used to turn this support on or off.

SYS

The SYS invocation option, in the form SYS=x, where x is an arbitrary character, identifies an alternate Cisco IOS for S/390 VTAM application to handle the Telnet connections established by the Client FTP2 program. This option is useful in cases where multiple copies of Cisco IOS for S/390 are running concurrently. You can specify SYS=x to access a network server called ACCESx instead of the usual ACCES.

WAIT

The WAIT option causes the terminal keyboard to remain locked while a data transfer is in progress. Normally the keyboard remains unlocked allowing additional commands to be entered during a data transfer.

This option is automatically set if the FIOS option is specified.

Client FTP2 Debugging Options

The Client FTP2 debugging options are used to gather debugging information on the internal operation of the Client FTP2 program and the interactions between the Client FTP2 program and the Server FTPs. Normally the debugging information displays on the terminal, but if the FIOS option is in effect, the information is written to the data set represented by the SYSPUT DD statement.

Note Use these options only under the direction of Cisco IOS for S/390 software support personnel.

TEST

Use the TEST option to display all requests sent and all responses received on the control connections to each Server FTP. This option can be turned on or off while the program is running by issuing the **DEBUG** command.

Information logged by the TEST option has this format:

TEST - *test debugging information text*

TESTI

Use the TESTI option to obtain local terminal input and output information, as well as to provide the TEST and DISP information.

Information logged by the TESTI option has this format:

TESTI - *testi debugging information text*

VLT

The VLT option turns on tracing of the virtual line terminal sessions associated with the FTP2 session. This option is useful for debugging VTAM problems between the FTP2 session and the Cisco IOS for S/390 address space. The VLT option generates an enormous amount of output. When used interactively, this output comes to the terminal. When used in batch, the output is written to the SYSVLT DD. When you run FTP2 in batch, you must add this DD statement to the JCL:

//sysvlt dd sysout=*,dcb=blksize=133

General Client FTP2 Operation

These steps outline the general procedure for using the Client FTP2 program:

- Issue the FTP2 command (with any optional parameters) to log on to the remote host. You are automatically logged on to the local MVS Cisco IOS for S/390 host.
- If you do not specify a HOST**=** option in your FTP2 command line, or if the remote host you specified is not in the NETRC file, you must issue an **OPEN** command for the remote Server FTP.
- If you specify a host name in your **OPEN** command and the host name is in the NETRC file, the user ID and password are taken from the NETRC file. If the host name is not in the NETRC file, you will be prompted for a user ID and password.

  If your user ID and password fail the logon because one or both were entered incorrectly, you must enter the **LOG** command to sign on to the remote host.
- Set the appropriate file transfer parameters (such as, MODE, STRUCT, or TYPE).
- Perform the desired transfer operation (such as, GET and PUT).

Path Name

A path name is a string that identifies a file to a file system. A path name must contain a device and/or directory name and a file name. The FTP specification does not specify a standard path name convention. Each user must follow the file naming conventions of the file systems involved in the transfer. Consult the System Administrators at the host sites involved in the transfer for file naming conventions.

Many of the Client FTP2 commands take one or more path name arguments. For information about the syntax for MVS path names supported by the Cisco IOS for S/390 Server FTP, read Data Set Names in Server FTP.

Client FTP2 Command Conventions

These general notes apply to the Client FTP2 commands:

Program Prompt

To indicate successful completion of most commands, the Client FTP2 program gives a new prompt. However, when a data transfer command is issued, a prompt appears when the operation begins successfully. You can then enter other commands (such as, status requests with the **STAT** command) while the operation proceeds.

Completion of Data Transfer

Final completion of the data transfer command is indicated with a message.

Step 1 Testing the Control Connections

You can use the TEST invocation option to see the specific FTP commands and responses sent and received over the control connections as a result of Client FTP2 commands. If you want more information about this test, refer to *RFC 959*.

Case Sensitivity

The Client FTP2 commands are not case sensitive.

Abbreviations in Commands

Abbreviations are permitted if they are not ambiguous. For example, you can type AB for ABORT but you cannot type only A because several commands begin with this letter.

Brackets in Commands

In the examples, parameters enclosed by brackets are optional for the command line. In many cases, if the optional parameters are omitted from the command line, you are prompted for them.

Syntax Conventions

Command words are shown in uppercase and parameters are shown in lowercase. When the actual values for a parameter are given (such as, **DEBUG[ON/OFF]**), they are shown in uppercase.

Example Conventions

In all examples of Client FTP2 input and output in this manual, user entries are shown in boldface type.

The Client FTP2 examples in this manual assume that you have issued **OPEN** and implied **LOG** commands similar to this example to connect an IBM MVS TSO user to another system:

```
OPEN unix
220 unix FTP server (SunOS 4.0) ready.
Enter name (unix:jim): jim
331 Enter PASS command
Password:
```

The NETRC File

The Client FTP2 program uses information in the NETRC file for automatic login to the remote host. The NETRC file is assumed to be named dsnpref.FTP.NETRC. The dsnpref is the same as the TSO profile prefix at the time of execution of the **FTP2** command. If noprefix is set in the TSO profile, then the dsnpref is the same as the executing TSO user ID. To use a different naming convention for this file, you must pre-allocate it with an NETRC DD statement.

You can specify any number of MACHINE/LOGIN/PASSWORD/ACCOUNT sets to define remote hosts. If you define a machine with no login, password, and/or account, you are prompted for this information when needed. When a user connects to a remote host, if a remote MACHINE/LOGIN/PASSWORD/ACCOUNT set exists for the host, this set logs the user on to the remote host.

The NETRC file also contains macros executed by the FTP2 program. You can define up to twenty macros in the NETRC file. Begin each macro with a MACDEF statement, enter valid Client FTP2 commands, and terminate the macro with an ENDMAC statement. If you define a macro named INIT, it is executed automatically before the Client FTP2 program issues the first prompt. (You do not need to issue a **$** command to execute the INIT macro.) For more information about the MACDEF statement, read MACDEF.

Note The OPEN statement in your SYSGET SYSIN must exactly match the corresponding machine statement in your NETRC. That is, host, port, and route must match. The INIT MACRO is executed prior to login to the remote host.

If you have not created a NETRC file, you are prompted to supply a user ID and password for the remote host whenever you open a connection to a remote host.

This is the format of the NETRC file; edit this file to specify user IDs, passwords, and accounts or to create a macro.

```
MACHINE remote_host_name
LOGIN userid (for preceding remote host)
PASSWORD password (for preceding remote host)
ACCOUNT account (for the preceding remote host)
MACDEF macro-name
stat
ENDMAC (terminates a macro)
```

If an installation does not require a password and/or an account, these can be omitted. Before using the following Client FTP2 commands, you must log on to a remote host with an **OPEN** command.

Caution You should use your local access control facility to protect NETRC files since these files can contain valid user ID password combinations for remote hosts. Only the TSO user ID using the NETRC file should be able to read their own NETRC file.

The NETRC file can also be used in batch jobs by using the NETRC invocation along with a NETRC DD pointing to your NETRC data set.

Note The batch invocation for NETRC does not assume a default FTP.NETRC data set name.

Client FTP2 High-Level Qualifier

The DEFPRFX parameter on the FTP statement in the file APPCFGxx is a server configuration parameter. It tells Server FTP what is the default high level qualifier to be used when starting an FTP server session. You can change the high level qualifier with the appropriate directory commands (**CWD** and **CDUP**).

The default high level qualifier used by Client FTP2 after automatic sign-on to the local MVS host usually matches what was specified on the DSNPRFX parameter.

Client FTP2 has a feature that extracts a TSO profile prefix when running under TSO (either interactively or in batch). If a user ID and TSO profile prefix (not set to NOPREFIX) are different, Client FTP2 issues a **CWD** command to change the default directory to the user's TSO profile prefix after the user connects and logs on to the local MVS host. This **CWD** command is not issued if the TSO profile prefix is the same as the TSO user ID or if the TSO profile is set to NOPREFIX.

User ID USER01 has a TSO profile prefix of XXYY. After automatically connecting to and signing on to the local MVS host, this server command is issued by FTP2 to the local MVS host:

CWD 'xxyy.'

Client FTP2 issuing the **CWD** command (when the TSO profile prefix is different from the user ID) may seem to conflict when a user sets DSNPRFX(NONE) on the FTP statement in member APPCFGxx. However, some users use this feature when running Client FTP2. Use this command for a simple work-around to remove the directory prefix when running under Client FTP2:

LQUOTE CDUP

Client FTP2 Commands

Table 2-1 gives a brief description of each command and its function; detailed descriptions of each command follow the table.

Table 2-1 Client FTP2 Commands

Command	Description
? [*command_name*]	Get help on all commands or one command
!*tso_command parameters*	Execute TSO command (same as **DO**)
$ *macro_name*	Execute a predefined list of commands
ABOR	Terminate data transfer immediately
ALLO *num_bytes* [**R** *size*]	**ALLO** provides a file size to those Server FTPs that require it. The Cisco IOS for S/390 Server FTP does not require use of the **ALLO** command; use it optionally to truncate records
APPEND [*local_path*] [*remote_path*]	Use **APPEND** to append records to a specified file on the remote host
BINARY	**BINARY** sets the file transfer mode to binary
BYE	Terminate program; same as **END** and **QUIT**
CD [*path*]	Change to the directory specified
CDUP	Change to parent of current working directory
CLOSE	Disconnect from remote host; same as **DISCONNECT**
DEBUG [**ON**|**OFF**]	Display all network commands and responses
DELETE [*path_name*]	Delete file on remote host
DIR [*remote_path*] [*local_path*]	Display contents of the directory specified on the remote host
DISCONNECT	Same as **CLOSE**
DO *tso_command parameters*	Run TSO command; same as !
EBCDIC	Set file transfer type to EBCDIC
END	Terminate program; same as **BYE** and **QUIT**
EXPE	Toggle experimental mode for directory commands
FIREWALL	Toggle implementation of Firewall-Friendly FTP
GET [*remote_path*] [*local_path*]	Copy file from remote host; same as **RECV**
HELP [*text*]	Display local host command information
LOG [*userid*] [*password*] [*/new_password*]	Log in user
LQUOTE [*text*]	Send an FTP command to the local server.
LS [*remote_path*] [*local_path*]	Display names of files in current working directory on remote host
MACDEF *macro_name*	Create a macro
MKDIR [*path_name*]	Create a new directory
MODE S|**B**	Set transmission mode
NTRANS [*inchars*] [*outchars*]	Translate file names
OPEN [*host_name*]	Open connection to remote host
PUT [*local_path*] [*remote_path*]	Copy file from local host to remote host; same as **SEND**
PWD	Show name of working directory on remote host

Table 2-1 Client FTP2 Commands (Continued)

Command	Description
QUIT	Terminate program; same as **BYE** and **END**
QUOTE [*text*]	Send FTP command to remote server
RECV [*remote_path*] [*local_path*]	Copy file from remote host to local host; same as **GET**
REMHELP [*text*]	Ask remote host for command information
REMSITE *text*	Send host-specific information to remote host
REMSNDS	Resend last **REMSITE** command to remote host
REMSTAT [*path_name*]	Display status of remote host
RENAME [*old_path_name*] [*new_path_name*]	Rename file on remote host
REST *local_marker remote_marker*	Restart
RMDIR [*path*]	Remove directory on remote host
SEND [*local_path*] [*remote_path*]	Copy file from local host to remote host; same as **PUT**
SITE *text*	Send information specific to local host
SNDS	Resend last **SITE** command to local host
STATUS [*parameter*]	Display status of local host
STRUCT F|**R**	Set file structure
SUNIQUE	Store unique file names on remote host
TYPE I | L *byte_size* | { **A** | **E** [**N** | **T** | **C**] }	Set data type

? Command

Use the **?** command to display information about using the Client FTP2 program.

? [*command_name*]

Syntax Description

command_name Command for which you are requesting information.

Default

If the **?** command is specified with no arguments, it displays a list of Client FTP2 commands.

Usage Guidelines

The **?** command can be used either with or without arguments.

If *command_name* is specified as an argument to the **?** command, a line of information displays similar to the information in the table in Client FTP2 Commands. The line shows the command syntax and a short description of the command function.

Example

```
? Dir
```
DIR <*pathr***> <***pathl***> - Directory list of remote host**

Related Commands

?	Requests help from the Client FTP2 program.
HELP	Requests help from your local Server FTP.
REMHELP	Requests help from the remote Server FTP.

! Command

The **!** command (do **TSO** command) requests the Client FTP2 program to execute a **TSO** command for you.

! *TSO_command parameters*

Syntax Description

tso_command parameters Lists the parameters to be passed to the **TSO** command.

Usage Guidelines

The *tso_command* argument is required.

This command requires that the TSO environment exist.

The **!** command cannot be used for batch FTP2 without the TMP.

Example

```
! listc
IN CATALOG: CATALOG.MVSICF1.VMVSTSO
USER1.ACCES.ASM
USER1.LIB.LOAD
USER1.T.D
USER1.VBIG.D
USER1.VB.D.
```

$ Command

The **$** command executes a specified macro you have defined either with a **MACDEF** command or in the NETRC file (see the sections of this chapter describing the **MACDEF** command and the NETRC file for more information).

$ *macro_name*

Example

In the following example, **$** is calling a macro called **SP**, which displays status, the current working directory, and ends the macro.

```
macdef sp
SP:stat
SP:pwd
SP:endmac
$ sp
EXECUTING: stat
Connected to: UNIX
--- STATUS ---
-- FTP Parameters ---
Remote DT Host, Port 127.0.0.1, 0
Local DT  Host, Port 138.42.224.15, 0
Type A N   Tabs 8   Stru F   Mode S   Recall 5
-- END --
-- Control --
User USER1   Acct Accs E0000200   Unit SYSALLDA
Host 138.42.224.15
-- End Control --
-- Path Data --
Rlse
-- End Path Data --
-- Transfer Information --
Data transfer not in progress
-- END --
211 <End of Status>
Executing: pwd
257 "/home/unix/user1" is current directory.
Executing: endmac
```

ABOR

Use **ABOR** (abort) to instruct the Server FTP to abort the last command issued and any associated transfer of data. No action is taken by the Server FTPs if the previous command has been completed (including data transfer). The control connections to the Server FTPs are not closed, but the data connections are closed.

ABOR

Syntax Description

This command has no arguments or keywords.

Default

The **ABOR** request is directed to both Server FTPs.

Example

```
get +outmail temp.data
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Path USER1.TEMP.DATA
Volser HAGCAT   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 3120   Space 1 5 Tracks Rlse
150 ASCII data connection for +outmail (138.42.224.15,4099) (167076 bytes)
ABOR
-Data transfer aborted.
 65536 bytes received in 6.65 seconds (9855 bytes/s)
Path USER1.TEMP.DATA   User USER1   Data bytes received 68948
Disk tracks written 1   Records padded 2997
226 Abort command completed.
426 Transfer aborted. Data connection closed.
226 Abort successful
225 ABOR command successful.
```

ALLO

The **ALLO** command allocates a file size to those Server FTPs that require it. Refer to the documentation for your particular Server FTP to see if you need this command. The Cisco IOS for S/390 Server FTP does not require use of the **ALLO** command; you can use it to cause records to be truncated or to allocate space.

ALLO *num_bytes* [**R** *size*]

Syntax Description

num_bytes	Number of bytes (using the logical byte size) of storage to be reserved for the file.
R *size*	Maximum size of storage required.

Usage Guidelines

If files are sent with record or page structure, a maximum record or page size (in logical bytes) can be required.

The R *size* argument is optional; if specified it must be separated from the first argument by the three characters, " R " (space R space)

The **ALLO** command requires a host prefix. For information on host prefixes, see Using Host Name Strings in Introduction to Cisco IOS for S/390.

Example

```
stru r
site lrecl(60) blk(6000)
allo 12000 r 60
put n.d allor.data
ALLOCMD = ALLO 12000 r 60
150-Data set open with attributes:
Type A N   Tabs 8   Stru R   Mode S   Recall Path USER1.ALLOR.DATA
Volser ICSPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space 1 1 Tracks Rlse   Maxr 60
-Data set open with attributes:
Type A N   Tabs 8   Stru R   Mode S   PATH USER1.N.D
Volser COLPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 3120   Rlse
226-Transfer complete
 3441 bytes received in 0.37 seconds (9300 bytes/s)
Path USER1.ALLOR.DATA   User USER1   Data bytes received 3277
Disk tracks written 1   Records truncated 21   Records padded 80
-Transfer complete
 3441 bytes sent in 0.25 seconds (13764 bytes/s)   Path USER1.N.D
User COLMBIA   Data bytes sent 6480
Disk tracks read 1
```

APPEND

The **APPEND** command requests that a file from the local host be appended to a file at the remote host.

APPEND [*local_path*] [*remote_path*]

Syntax Description

local_path	Name of the file to be retrieved from the local host.
remote_path	Remote file to which the local file is to be appended.

Default

If either file name is omitted, you are prompted for the file name.

Usage Guidelines

The syntax for each path depends on the associated Server FTP.

Example

```
append n.d myappend.data
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Path USER1.N.D
Volser COLPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 3120   Rlse
150-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.MYAPPEND.DATA
Volser HAGCAT   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space 5 3 Tracks Rlse
-Transfer complete
 3439 bytes sent in 0.46 seconds (7476 bytes/s)   Path USER1.N.D
User COLMBIA   Data bytes sent 6480
Disk tracks read 1
226-Transfer complete
 3439 bytes received in 0.42 seconds (8188 bytes/s)
Path USER1.MYAPPEND.DATA   User USER1   Data bytes received 3277
Disk tracks written 1   Records padded 80
```

ASCII

The **ASCII** command sets the data type to ASCII for the transfer of an ASCII file.

ASCII

Example

```
ASCII
stat
Connected to: MVS
--- STATUS ---
--- FTP PARAMETERS ---
Remote DT Host, Port 138.42.224.13, 0
Local DT  Host, Port 138.42.224.15, 0
Type A N   Tabs 8   Stru F   Mode S   Recall 5  Server is passive
-- END --
-- Control --
User COLMBIA   Acct Accs E0000200   Unit SYSALLDA   Host 138.42.224.15
-- End Control --
-- Path Data --
Rlse
-- End Path Data --
-- Transfer Information --
Data transfer not in progress   Data bytes sent 6480
Disk tracks read 1   Network bytes sent 3439   Elapsed time 00.00.00
Bytes/Second 7476
-- END --
211 <End of Status>
```

Related Commands

TYPE	This command is equivalent to a **TYPE** command with the ASCII (**A**) parameter specified.

BINARY

The **BINARY** command sets the data type to binary for the transfer of a binary file.

BINARY

Syntax Description

This command has no arguments or keywords.

Example

```
BINARY
stat
Connected to: MVS
--- STATUS ---
-- FTP Parameters --
Remote DT Host, Port 138.42.128.13, 0
Local DT  Host, Port 138.42.224.15, 0
Type I N   Stru F   Mode S   Recall 5   Server is passive
-- END --
-- Control --
User COLMBIA   Acct Accs E0000200   Unit SYSALLDA   Host 138.42.224.15
-- End Control --
-- Path Data --
Rlse
-- End Path Data --
-- Transfer Information --
Data transfer not in progress   Data bytes sent 6480
Disk tracks read 1   Network bytes sent 3439   Elapsed time 00.00.00
Bytes/Second 7476
-- END --
211 <End of Status>
```

Related Commands

TYPE	This command is equivalent to a **TYPE** command with the image (**I**) parameter specified.

BYE

The **BYE** command terminates the Client FTP2 program. This command is the same as the **END** and **QUIT** commands.

BYE

Syntax Description

This command has no arguments or keywords.

Example

```
BYE
221 Goodbye.
```

Related Commands

END	Can be used instead of **BYE**; **END** does not require a host prefix.
QUIT	Can be used instead of **BYE**; **QUIT** requires a host prefix and takes no arguments.

CD

The **CD** command requests that the remote Server FTP change the current directory to a new directory.

CD [*path_name*]

Syntax Description

path_name Indicates to the remote Server FTP the name of the directory to be made the current directory.

Default

If you omit *path_name*, the Client FTP2 program prompts you for the desired value.

Usage Guidelines

The syntax for *path_name* depends on the associated Server FTP.

Example

A UNIX Server FTP in session with the Client FTP2 program is using /u/user1/work as the current directory. If a **CD JUNK** command is issued by the Client FTP2 to that UNIX Server FTP, the resulting current directory is /u/user1/work/junk. The same result is achieved by specifying **CD** /u/user1/work/junk.

This example shows a change of directory to a remote UNIX system:

CD /u/lpn/d.ddn

```
250 CWD command successful.
```

This example shows a change to a directory on a remote system running Cisco IOS for S/390:

CD acces

```
250 "'USER1.ACCES.'" is current prefix
```

CDUP

The **CDUP** command directs the remote Server FTP to change the current directory to the parent directory of the old current directory. The **CDUP** command is most useful when the Server FTP manipulates a hierarchical file system such as UNIX.

CDUP

Syntax Description

This command has no arguments or keywords.

Example

A UNIX Server FTP in session with the Client FTP2 program has */u/user1/work* as the current directory. If a **CDUP** command is issued by the Client FTP2 to that UNIX Server FTP, the resulting current directory is the parent of the old current directory (*/u/user1*).

This example shows a change to the parent directory of the old current directory on a remote UNIX system:

```
CDUP
250 CWD command successful.
```

CLOSE

The **CLOSE** command logs you out and terminates the connection between you and the remote Server FTP.

CLOSE

Syntax Description

This command has no arguments or keywords.

Example

```
CLOSE
221 Goodbye.
```

DEBUG

The **DEBUG** command displays all commands and responses going to and from the Server FTPs and the user. This command is equivalent to specifying the TEST option on the Client FTP2 command line.

DEBUG [ON | OFF]

Syntax Description

DEBUG ON Enables you to see all commands and responses going to and from the Server FTPs.

DEBUG OFF Turns off the DEBUG option.

Default

If no parameter is specified with the **DEBUG** command, the command toggles the previous state.

Example

```
DEBUG ON
```

DELETE

The **DELETE** command directs the remote Server FTP to delete the specified file.

DELETE [*path_name*]

Syntax Description

path_name Specifies the specific file to delete.

Default

If you omit *path_name*, you are prompted to supply one.

Usage Guidelines

The syntax for *path_name* depends on the associated Server FTP.

Example

```
DELETE gatedel
250 DELE command successful.
```

DIR

The **DIR** command requests that the remote Server FTP provide a directory list for the specified path.

DIR [*remote_path*] [*local_path*]

Syntax Description

remote_path specifies the remote directory to be listed. If *remote_path* is not specified, you receive a list of your current directory

local_path specifies the file to which the directory list is written. If *local_path* is not specified, the directory list is displayed on your screen

Defaults

The current directory is displayed to your screen.

Usage Notes:

The **DIR** command requires that the TSO environment exist (that is, you are not running batch FTP2 without the TMP).

Example

The following example lists the contents of a directory on an MVS system.

```
ftp> DIR /export/home/user1/mvs
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.FTP.TMP.T1330492
Volser ICSUSR   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space 5 3 Tracks Rlse
150 ASCII data connection for /bin/ls (138.42.224.15,4126) (0 bytes).
226 ASCII Transfer complete.
-Transfer complete
 322 bytes received in 1.50 seconds (214 bytes/s)
Path USER1.FTP.TMP.T1330492   User USER1   Data bytes received 310
Disk tracks written 1   Records padded 6
total 20
-rw-------     1 user1     dvlp     2730 Oct 22     08:36 channel
-rw-------     1 user1     dvlp     2901 Sep  1     09:17 comten
-rw-------     1 user1     dvlp     1276 Oct 21     14:43 dump
-rw-------     1 user1     dvlp      729 Nov 12     05:24 ibmlink
-rw-------     1 user1     dvlp      751 Nov 12     07:41 ibmlink2
IDC05501 ENTRY (A) USER1.FTP.TMP.T1330492 DELETED
ftp>
```

DISCONNECT

The **DISCONNECT** command disconnects you from a host. It logs you out and terminates the connection between you and the remote Server FTP. The CLOSE command is the same as the **DISCONNECT** command.

DISCONNECT

Syntax Description

This command has no arguments or keywords.

Example

The following example shows the **DISCONNECT** command.

```
ftp> DISCONNECT

disconnect
221 Goodbye.
ftp.
```

DO

The **DO** (do TSO command) command requests the Client FTP2 program to execute a TSO command for you.

DO *tso_command parameters*

Syntax Description

tso_command parameters Specifies the TSO command followed by any parameters to be passed to the TSO command.

Usage Notes

The *tso_command* argument is required.

The **DO** command is handled by the Client FTP2 program.

The **DO** command requires that the TSO environment exist (in other words, you are not running batch FTP2 without the TMP).

Example

```
DO listc l(lpn)
NONVSAM-------LPN.ABOR
IN-CAT--CATALOG.MVSICF1.VMVSTSO
NONVSAM-------LPN.FTPNETRC
IN-CAT--CATALOG.MVSICF1.VMVSTSO
NONVSAM-------LPN.CNTL
IN-CAT--CATALOG.MVSICF1.VMVSTSO
NONVSAM-------LPN.FTPEXAMPLE
IN-CAT--CATALOG.MVSICF1.VMVSTSO.
```

EBCDIC

The **EBCDIC** command sets the data type to EBCDIC for transfer of an EBCDIC file. The command is equivalent to the **TYPE** command with the EBCDIC (**E**) parameter specified.

Example

```
EBCDIC
EBCD ENTERED
stat
CONNECTED TO: MVS
--- FTP PARAMETERS ---
REMOTE DT HOST,PORT 129.192.224.136, 0
LOCAL DT HOST,PORT 129.192.224.136, 20
TYPE E N TABS 8 STRU F MODE S
--- ACCESS CONTROL ---
USER LPN ACCT ACCS E0000200 VOLS MVSTSO TELNET HOST 129.192.224.136
SESSION# 412
--- PATH DATA ---
--- TRANSFER INFORMATION ---
DATA TRANSFER NOT IN PROGRESS
```

END

The **END** command terminates the Client FTP2 program. This is typically the last command you enter. Any open control connections are closed before the program terminates.

END

Syntax Description

This command has no arguments or keywords.

Example

```
END
221 Session terminated
```

Related Commands

BYE	Can be used instead of **END**. **BYE** works exactly as the **QUIT** command.
QUIT	Can be used instead of **END**; **QUIT** requires a host prefix and takes no arguments.

EXPE

The **EXPE** command toggles the use of experimental or regular directory commands. Since there is no consistent support for this command, it is recommended that you not use this command.

EXPE

Defaults

No operands are associated with the **EXPE** command since it is a Client FTP2 command.

Usage Guidelines

Table 2-2 shows the FTP command that is sent over the control connection for each directory command with an **EXPE** setting:

Table 2-2 Using Directory Commands with an EXPE Setting

Client FTP2 Command	Regular	Experimental
MKD	**MKD**	**XMKD**
RMD	**RMD**	**XRMD**
PWD	**PWD**	**XPWD**
CDUP	**CDUP**	**XCUP**

Note The directory commands were added to FTP subsequent to the initial FTP specification and are documented in *RFC 959*, File Transfer Protocol (FTP) Appendix II, Directory Commands.

FIREWALL

The **FIREWALL** command toggles the implementation of Firewall-Friendly FTP RFC **1579** on and off. RFC **1579** specifies that for FTP data connection establishment, the client sends the **PASV** command to the remote and the **PORT** command to the local host. This is the default. When the **FIREWALL** command is entered to disable this implementation, the FTP2 client will send the **PASV** command to the local and the **PORT** command to the remote. You can turn the implementation back on by entering **FIREWALL** again. A message is returned to the user indicating if Firewall is enabled or disabled.

FIREWALL

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

No arguments are associated with the **FIREWALL** command.

Note This option may also be turned off via the NOFIRE parameter of the Client FTP2 Invocation Options

---

Example

The following example shows the effect of turning the firewall on and off. Initially, the firewall implementation is on.

```
10:06:23   FTP2:  put n.d temp
10:07:26     TEST - SENDING - B:PASV
10:07:27   227 Entering Passive Mode (138,42,32,165,145,225)
10:07:27     TEST - SENDING - A:PORT 138,42,32,165,145,225
10:07:28   200 OK, Ready
10:07:28     TEST - SENDING - A:RETR n.d
10:07:29   150-Dataset open with attributes:
10:07:29   Type A N    Tabs    8   Stru F   Mode S   Path ABC.N.D   Volser ICS007
10:07:29   Unit SYSICS   Dsorg          PS   Recfm FB   Lrecl   80   Blksize   3120
10:07:29   Rlse
10:07:30   150
10:07:30     TEST - SENDING - B:STOR temp
10:07:31   150 ASCII data connection for temp (138.42.220.13,20).
10:07:31   FTP2:
10:07:34   226-Transfer complete
10:07:35    3439 bytes sent in 1.17 seconds (2939 bytes/s)   Path ABC.N.D
10:07:35   User ABC Data bytes read 6480
10:07:36   Disk tracks read 1
10:07:36   226
10:07:36   226 Transfer complete.
10:07:36   FTP2:  firewall
10:10:28   ACC831I - Firewall has been disabled
10:10:28   FTP2:  put n.d temp
10:10:58     TEST - SENDING - A:PASV
10:11:00   227 Entering passive mode    138,42,220,13,20,27
10:11:00     TEST - SENDING - B:PORT 138,42,220,13,20,27
10:11:01   200 PORT command successful.
10:11:01     TEST - SENDING - A:RETR n.d
10:11:02   150-Dataset open with attributes:
10:11:03   Type A N    Tabs    8   Stru F   Mode S   Path ABC.N.D   Volser ICS007
10:11:04   Unit SYSICS   Dsorg          PS   Recfm FB   Lrecl   80   Blksize   3120
10:11:04   Rlse
10:11:04   150
10:11:04     TEST - SENDING - B:STOR temp
10:11:05   150 ASCII data connection for temp (138.42.220.13,5147).
10:11:05   FTP2:
10:11:27   226-Transfer complete
10:11:28    3439 bytes sent in 1.22 seconds (2818 bytes/s)   Path ABC.N.D
10:11:28   User ABC   Data bytes read 6480
10:11:28   Disk tracks read 1
10:11:28   226
10:11:28   226 Transfer complete.
10:11:28   FTP2:  firewall
10:13:10   ACC831I - Firewall has been enabled
10:13:10   FTP2:
```

GET

The **GET** command requests that a file from the remote host be copied to a file on the local host by the appropriate Server FTPs. The file to be retrieved is always at the remote host, and the file to be copied into is always at the local host.

GET [*remote_path*] [*local_path*]

Syntax Description

remote_path Specifies the file name to be retrieved from the remote host.

local_path specifies the file name at the local host into which the file from the remote host is copied.

Defaults

If you omit either file name, you are prompted for one.

Usage Guidelines

The syntax for each path depends on the associated Server FTP.

Note The **FTP GET DGDBASE** command gets only the most recent data set. Check with your Cisco IOS for S/390 Administrator to find out if existing data sets are allowed to be overwritten (this is specified on the FTP statement in APPCFG*xx*).

Example

The following example gets the file comten from the remote host and saves it on the local host as temp.data:

```
ftp> GET comten temp.data
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S  Recall 5  Path USER1.TEMP.DATA
Volser HAGCAT   Unit SYSALLDA   Dsorg  PS   Recfm FB   Lrecl  80
Blksize 3120   Space 1 5 Tracks Rlse
150 ASCII data connection for comten (138.42.224.15,4127) (2091 bytes).
-Transfer complete
 2291 bytes received in 1.34 seconds (2232 bytes/s) Path USER1.TEMP.DATA
User USER1   Data bytes received 2825
Disk tracks written 1   Records padded 90   Records folded 2
226 ASCII Transfer complete.
ftp>
```

Related Commands

RECV This command is the same as the **GET** command.

HELP

The **HELP** command requests help information from the local Server FTP.

HELP [*text*]

Syntax Description

The **HELP** command has no required operands. Any operands specified on the **HELP** command are passed through unchanged to the Server FTP and are interpreted by the Server FTP.

Usage Guidelines

After a **REST** command, **STOR** and **APPE** have identical meanings.

Data transfer must be MODE B (block).

A file retrieved normally includes restart markers approximately every 32767 bytes. The **REST** parameter on the **SITE** command allows the user to change this interval or even entirely suppress restart markers. See **HELP SITE**. The actual decision to send a marker depends on a count of data bytes read from the disk (not including OS count/control bytes). When this count reaches the limit, the marker is sent at the next end of a complete logical record, segment of a spanned record (if **RECFM** includes **VS**), or a physical disk block (if RECFM is U, V, or F).

FTP can accept (and send) restart markers in either STRU F or STRU R.

FTP restart markers consist of 10 characters, which are the hexadecimal representation of five 8-bit bytes: *TTRBB*. Here, *TTR* forms a standard OS disk block address, and *BB* is a byte offset within the block.

Example

```
HELP REST
--- HELP---
*** HELP REST ***
FTP REST (Restart) Command:
Function: Specifies that the data transfer command that follows  (immediately) is to
restart
at a specified intermediate point in the file.
Syntax:  REST <marker>
214  <end of HELP>
```

Related Commands

? Command	Requests help from the Client FTP2 program.
HELP	Requests help from your local Server FTP.
REMHELP	Requests help from the remote Server FTP.

LOG

The **LOG** (login) command gives a user ID and password to a remote Server FTP to identify the user. You can optionally change your password when logging in to the Cisco IOS for S/390 server. The **LOG** command typically is issued immediately following the **OPEN** command.

LOG [*userid*] [*current_password*] [*/new_password*]

Syntax Description

userid	User name.
current_password	Password for the user name.
new_password	New password if you choose to change passwords upon logging in.

Default

The Client FTP2 program prompts you for the user ID and/or current password.

Usage Guidelines

If the remote Server FTP requires additional accounting information during the user identification process, the Client FTP2 program prompts you to enter the accounting data.

Examples

The /new_password parameter is a 1- to 8-character string password. The new password replaces the current password after the user ID and current password are validated. The new password option is valid only when talking to the Cisco IOS for S/390 server. The slash (/) must follow the current password without any intervening blanks. The new password must follow the slash without any intervening blanks, as in the following example:

```
LOG lpn tstpass
230 User lpn logged in.
```

In the following example, user USER01 wants to change his current password from CJAY to MACDUFF.

```
LOG user01 cjay/macduff
230 User USER01 logged in.
```

LQUOTE

The **LQUOTE** command sends an uninterpreted, unaltered character string to the local Server FTP over the control connection. This mechanism sends FTP commands to the Server that the Client FTP2 program might not be able to send.

LQUOTE [*text*]

Syntax Description

This command has no arguments or keywords.

Default

If the text is omitted, you are prompted for it.

Usage Guidelines

The text is sent to the local FTP Server over the control connection exactly as you entered it.

Example

```
LQUOTE site vol(mvsts2)
```

LS

The **LS** command requests a remote Server FTP to provide a list of file names for the specified path.

LS [*remote_path*] [*local_path*]

Syntax Description

remote_path Path to be listed from the remote host.

local_path Local file into which the list from the remote server is printed.

Default

If a local path is not specified, the list of files appears on your screen.

Usage Guidelines

If the path specifies a directory or other group of files, the remote Server FTP transfers a list of files.

The syntax for each path depends on the associated Server FTP

If a local path is specified, the list of files is written to the specified file.

The **LS** command requires that the TSO environment exist (that is, you are not running batch FTP2 without the TMP).

Examples

The following example shows the **LS** command with parameters:

```
LS /export/home/user1 unix.dir.temp
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.UNIX.DIR.TEMP
Volser ICSUSR   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl  80
Blksize 6160   Space 5 3 Tracks Rlse
150 ASCII data connection for /bin/ls (138.42.224.15,4134) (0 bytes).
226 ASCII Transfer complete.
-Transfer Complete
 350 bytes received in 1.54 seconds (227 bytes/s)
Path USER1.UNIX.DIR.TEMP   User USER1   Data bytes received 274
Disk tracks written 1   Records padded 38
```

The following example shows the **LS** command without parameters:

```
LS
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.FTP.TMP.T1443273
Volser ICSUSR   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl  80
Blksize 6160   Space 5 3 Tracks Rlse
150 ASCII data connection for /bin/ls (138.42.224.15,4135) (0 bytes).
226 ASCII Transfer complete.
-Transfer complete
 42 bytes received in 1.47 seconds (28 bytes/s)
Path USER1.FTP.TMP.T1443273   User USER1   Data bytes received 32
Disk tracks written 1   Records padded 5
channel
comten
dump
filea
tempfile
IDC0550I ENTRY (A) USER1.FTP.TMP.T1443273 DELETED
```

MACDEF

The **MACDEF** command enables you to define a sequence of commands you want to execute more than once (a macro).

MACDEF *macro_name*

Syntax Description

macro_name Name of the macro being defined.

Usage Guidelines

When you issue the **MACDEF** command, you are prompted for each line of input. You must enter valid Client FTP2 commands.

To terminate input, press **RETURN**.

To execute the macro, issue a **$** command (read $ Command).

The macro is temporary and is deleted automatically at the end of your session. To create a permanent macro, read The NETRC File.

MACDEF handles \ (backslash) and $ (dollar) as special characters.

A $ followed by a digit 1-9 is replaced by the corresponding argument on the macro invocation command line (the $ command).

A \ followed by any character is replaced by that character. Use \ with $ to prevent special handling of the $.

Use \\ to cause a single \ to be interpreted.

Macros cannot be nested (one macro cannot call another macro).

Note A **NETRC** file must be used with the following example for login to the remote host.

Example

```
USERFTP: MACDEF getfrom
GET:open $1
GET:pwd
GET:get $2 $3
GET:close
GET:endmac
USERFTP: $ getfrom host1.md.company.com ftpdata 'abc.ftp.test'
Executing: open host.md.company.com
220 host FTP server (UNIX(r) System V Release 4.0) ready.
230 User abc logged in.
Executing: pwd
257 "/opt/home/abc" is current directory.
Executing: get ftpdata 'abc.ftp.test'
-Dataset open with attributes:
Type A N  Tabs  8   Stru F   Mode S   Recall 5   Path ABC.FTP.TEST
Volser ICSPK3  Unit SYSALLDA  Dsorg  PS   Recfm U
Blksize   6160    Space 1     Tracks   Rlse
150 ASCII data connection for ftpdata (138.42.128.13,4679)
   (2893 bytes
226 ASCII Transfer complete.
-Transfer complete
2927 bytes received in 1.16 seconds (2523 bytes/s)
  Path  ABC.FTP.TES
User ABC  Data bytes written 2927
Disk tracks written 1
Executing: close
Executing: endmac
```

MKDIR

The **MKDIR** (make directory) command directs a remote Server FTP to create the specified directory.

MKDIR [*path_name*]

Syntax Description

path_name Directory to be created.

Defaults

If you omit *path_name*, you are prompted for it.

Usage Guidelines

If the path name is relative, the specified subdirectory is created in the current working directory.

If the path name is absolute, the specified directory is created.

The syntax for path depends on the associated Server FTP.

Example

A UNIX Server FTP in session with the Client FTP2 program has */u/user1/work* as the current directory. If a **MKDIR junk** command is issued by the Client FTP2 to that UNIX Server FTP, the subdirectory junk is created in the current directory (*/u/user1/work/junk*). The same result is achieved by specifying **MKDIR** */u/user1/work/junk*.

```
MKDIR /u/lpn/d.new
257 MKD command successful.
```

MODE

The **MODE** command sets one of two transmission modes:

- Block mode

Block mode formats the data and allows for restart procedures.

- Stream mode

Stream mode passes the data with little or no processing. It interacts with the structure attribute to determine the type of processing. Stream mode is the default if no **MODE** command was used.

For the purpose of standardized transfer, the sending host translates its internal end-of-line or end-of-record representation into the representation required by the transfer mode and file structure, and the receiving host performs the inverse translation to its internal representation. Since these transformations make extra work for some systems, identical systems transferring non-record structured text files might use binary representation and stream mode to simplify transfer.

MODE S | B

One of the two codes (either s or b) is required as an argument on the **MODE** command.

Each of the possible transmission modes is discussed in the following sections. For a detailed description of the effect of various transmission modes, read the Transmission Modes section in *RFC 959*, File Transfer Protocol.

Not all Server FTPs support all transmission modes; review the Server FTP documentation if you have questions concerning transmission mode support.

Block Mode

Block mode is indicated with the character **B**. In block mode, the file is transmitted as a series of data blocks preceded by one or more header bytes. Record structures are allowed in this mode, and any representation type can be used. Restart markers are embedded in the data stream.

Stream Mode

Stream mode is set with the character **S**. This is the default if no **MODE** command has been used. In stream mode, the data is transmitted as a stream of bytes. There are no restrictions on the representation type used, and record structures are allowed. In a record structured file, End of Record (EOR) and End of File (EOF) are each indicated by a two-byte control code included with the data sent over the data connection. If the structure is a file structure, the EOF is indicated by the sending host closing the data connection, and all bytes sent over the data connection are data bytes.

NTRANS

The **NTRANS** command is executed prior to a file transfer to enable you to change the name of the file you are transferring. When you execute the file transfer and create a file at a destination, the transferred file has the name you specify with the **NTRANS** command.

NTRANS [*inchars*] [*outchars*]

Syntax Description

inchars	Specifies the characters in the file name that you want to change.
outchars	Specifies the new characters.

Default

If either parameter is omitted, you are prompted for it.

Example

Suppose that you have a file named ABC(DEF) and you issue this command:

```
NTRANS () .z
```

To effect the change specified in the **NTRANS** command, issue a **PUT** or **GET** command to transfer the file, as shown here:

```
PUT abc(def)
```

Issue the file transfer command without a second file name. The file ABC(DEF) is created at its destination with the name ABC.DEFZ (the left parenthesis is changed to a period (.) and the right parenthesis is changed to a Z).

```
NTRANS nd ab
PUT n.d
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Path USER1.N.D
Volser COLPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 3120 Rlse
150-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5   Path USER1.A.B
Volser ICSPK2   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space 5 3 Tracks Rlse
-Transfer complete
 3439 bytes sent in 0.55 seconds (6252 bytes/s)   Path USER1.N.D
User USER1   Data bytes sent 6480
Disk tracks read 1
226-Transfer complete
 3439 bytes received in 0.48 seconds (7164 bytes/s) Path USER1.A.B
User USER1   Data bytes received 3277
Disk tracks written 1   Records padded 80
```

OPEN

The **OPEN** command sets up one control connection between the Client FTP2 program and a Server FTP. You can also connect to a "remote" Server FTP that is actually your local Server FTP by specifying your own local host name.

OPEN [*host_name*]

Syntax Description

host_name	Name of the remote host.

Default

If you omit the host name, you are prompted for one.

Usage Guidelines

Host name strings must correspond to the syntax specified in Introduction to Cisco IOS for S/390.

Once a host connection is established, FTP2 prompts for a user ID, password (and optionally an account) combination when one is not provided in the NETRC file.

A successful **OPEN** command implies an automatic **LOG** command attempt by FTP2.

Example

```
OPEN unix
220 unix FTP server (SunOS 4.0) ready.
Enter name (unix:user1): user1
331 Enter PASS command
Password:
```

PUT

The **PUT** command requests that the appropriate local Server FTP copy a file to the remote system. The file to be copied is always at the local Server and the file destination is always at the remote Server.

PUT [*local_path*] [*remote_path*]

Syntax Description

local_path File name of the file to be copied from the local server.

remote_path File name of the file at the remote Server into which the file from the local server is copied.

Default

If you omit either path, you are prompted for the file name.

Usage Guidelines

The command name **SEND** can be used in place of **PUT**. There are no other differences when using **SEND**.

The syntax for each path depends on the associated Server FTP.

Example

```
PUT telnet.data teltemp
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Path USER1.TELNET.DATA
Volser COLPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 120
Blksize 3600   Rlse
150 ASCII data connection for teltemp (138.42.224.15,4104)
-Transfer complete
 11535 bytes sent in 2.22 seconds (5195 bytes/s)
Path USER1.TELNET.DATA
User COLMBIA   Data bytes sent 19680
Disk tracks read 1
226 ASCII Transfer complete.
```

PWD

The **PWD** command directs a remote Server FTP to return the path name of the current working directory.

PWD

Syntax Description

This command has no arguments or keywords.

Example

```
PWD
257 "/u/lpn" is current directory.
```

QUIT

The **QUIT** command terminates the Client FTP2 program. This command is the same as the **BYE** and **END** commands.

QUIT

Syntax Description

This command has no arguments or keywords.

Example

```
quit
221 Goodbye.
```

Related Commands

BYE	Can be used instead of **QUIT**. **BYE** works exactly as the **QUIT** command.
END	Can be used instead of **QUIT**; **END** does not require a host prefix.

QUOTE

The **QUOTE** command sends an uninterpreted, unaltered character string to the remote Server FTP over the control connection. This mechanism sends FTP commands to the Server that the Client FTP2 program might not be able to send.

QUOTE [*text*]

Syntax Description

text	Text sent to the Server over the control connection exactly as you enter it.

Default

If the text is omitted, you are prompted to enter it.

Example

```
QUOTE pasv
227 Entering Passive Mode (26,131,0,17,4,216).
```

RECV

The **RECV** command requests that a file from the remote host be copied to a file on the local host by the appropriate Server FTPs. The file to be retrieved is always at the remote host, and the file to be copied to is always at the local host.

RECV [*remote_path*] [*local_path*]

Syntax Description

remote_path File name to be retrieved from the remote host.

local_path File name at the local host that the file from the remote host is copied into.

Default

If you omit either file name, you are prompted for it.

Usage Guidelines

The syntax for each path depends on the associated Server FTP.

Note Check with your Cisco IOS for S/390 Administrator to find out if existing data sets are allowed to be overwritten (this is specified on the FTP statement in APPCFG*xx*).

Example

```
RECV teldata temp.data
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5   Path USER1.TEMP.DATA
Volser ICSUSR   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space 5 3 Tracks Rlse
150 ASCII data connection for teldata (138.42.224.15,4112) (11371 bytes).
226 ASCII Transfer complete.
-Transfer Complete
 11535 bytes received in 1.56 seconds (7394 bytes/s)
Path USER1.TEMP.DATA   User USER1   Data bytes received 11207
Disk tracks written 1   Records padded 147   Records folded 69
```

Related Commands

GET The **RECV** command is the same as the **GET** command.

REMHELP

The **REMHELP** command requests help information from the remote Server FTP

REMHELP [*text*]

Syntax Description

text	Any text specified on the **REMHELP** command is passed through unchanged to the remote Server FTP and is interpreted by the remote Server FTP.

Usage Guidelines

No operands are required for the **REMHELP** command.

The **REMHELP** command is different from the **?** command in that the **REMHELP** command requests help from the remote Server FTP, whereas the **?** command requests help from the Client FTP2 program.

Example

```
REMHELP list
214 Syntax: LIST .  <sp> path-name .
```

REMSITE

The **REMSITE** (remote site parameters) command provides the remote Server FTP with specific information it requires. This information is essential to file transfers involving that Server FTP, but is not sufficiently universal to have been included specifically in the FTP. Typically, you use a **REMHELP SITE** Client FTP2 command to find the site requirements for a specific remote Server FTP. Otherwise, review the Server FTP documentation for the site requirements.

REMSITE *text*

Syntax Description

text Text that is required and is passed through unchanged to the specified server.

Example

The following example shows a **REMSITE** command to change the default FTP volume:

```
REMSITE vol(mvsts2)
```

Note The *vol* parameter is specific to an MVS remote server.

REMSNDS

The **REMSNDS** (resend **REMSITE** parameters) command directs the Client FTP2 program to resend the last **REMSITE** command to the remote Server FTP. Your **REMSITE** command is reissued without your having to retype it. Since most Server FTPs require that new site parameters be provided before each data transfer, the **REMSNDS** saves time if you repeatedly use identical site parameters.

REMSNDS

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

You can preserve **SITE** parameters between data transfers by using the PERSIST option in the **SITE** command.

Example

```
SITE vol(mvsts2)
REMSNDS
site vol(mvsts2) <Sent
```

REMSTAT

The **REMSTAT** command requests a status response from the remote system.

REMSTAT [*path_name*]

Syntax Description

path_name Path on the remote host from which status is requested.

Default

If no path name is given, the remote Server FTP sends status information relative to parameters, connection status.

Usage Guidelines

The *path_name* argument is optional.

When **REMSTAT** is issued between data transfer operations, the *path_name* argument can be given.

With **REMSTAT**, the information is transferred over the control connection instead of the data connection.

The syntax of *path_name* depends on the associated Server FTP

The Cisco IOS for S/390 Server FTP program implements some additional parameters on the **remstat** command. Use a **HELP REMSTAT** Client FTP2 command to find additional parameters.

Example

```
REMSTAT /u/lpn
502 STAT command not implemented.
```

RENAME

The **RENAME** command directs a remote Server FTP to rename a file.

RENAME [*old_path_name*] [*new_path_name* **]**

Syntax Description

old_path_name File name to be renamed.

new_path_name New name to be assigned to that file.

Default

If you omit either argument, you are prompted to enter it.

Usage Guidelines

The syntax of the path names depends on the associated Server FTP.

Example

```
RENAME titlecol coltitle
350 File exists, ready for destination name
250 RNTO command successful.
```

REST

The **REST** (restart) command shows the Server FTP the restart marker where a file transfer is to be restarted. This command does not cause a file transfer but instead causes the Server FTP to skip over the file to the specified data checkpoint. This command should be followed immediately by the Client FTP2 command that causes the file transfer to resume.

Note The restart facility requires that you run in Mode B. Very few UNIX implementations support Mode B and these are unable to use the restart facility.

REST *local_marker remote_marker*

Syntax Description

local_marker Local marker where the restart is to begin.

remote_marker Remote marker from which the restart is to begin.

Usage Guidelines

Both marker arguments are required and represent the Server FTP marker where the file transfer is to be restarted.

The format of the restart marker is determined by the sending Server FTP and should be entered exactly as displayed during the interrupted file transfer.

Example

The following example shows a restart marker message received by a user during a previous file transfer:

```
B:110 MARK 010023010E8C = 010023010E8C
```

To restart the file transfer at the restart markers, issue the commands shown in bold in this output display:

```
MODE B
BINARY
SITE REST(100000)
REST 010023010E8C 010023010E8C
PUT 'scm.p016572.T01TCP' job16572.MYFILE
350 Requested file action pending further information
350 Requested file action pending further information
150-Data set open with attributes:
Type I N   Stru F   Mode B   Recall 5   Path USER1.JOB16572.MYFILE
Volser ICSPK2   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 133
Blksize 6650   Space 3 1 Cyl Rlse   Restart at 010023010E8C
-Data set open with attributes:
Type I N   Stru F   Mode B   Path SCM.P016572.MYFILE
Volser HAGCAT   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 133
Blksize 6650  Rlse  Bytes/Restart 100000   Restart at 010023010E8C
110 MARK 010025040F96 = 010025040F96
110 MARK 0100280110A0 = 0100280110A0
-Transfer complete
 223966 bytes sent in 1.91 seconds (117259 bytes/s)
Path SCM.P016572.MYFILE   User COLMBIA   Data bytes sent 218918
Disk tracks read 6   Restart markers send 2
226-Transfer Complete
 223966 bytes received in 1.90 seconds (117876 bytes/s)
Path USER1.JOB16572.MYFILE   User USER1   Data bytes received 218918
Disk tracks written 6   Records folded 1646
Restart markers received 2
```

RMDIR

The **RMDIR** (remove directory) command directs a remote Server FTP to remove the specified directory.

RMDIR [*path_name*]

Syntax Description

path_name Directory to be removed.

Default

If you omit *path_name*, you are prompted for it.

Usage Guidelines

If the path name is relative, the specified subdirectory is removed from the current working directory.

If the path name is absolute, the specified directory is removed.

The syntax of *path_name* depends on the associated Server FTP.

Note Many systems require the directory to be empty before it can be removed.

Example

As an example, if a UNIX Server FTP in session with the Client FTP2 program has */u/user1/work* as the current directory, and a **RMDIR** *junk* command is issued by Client FTP2 to that UNIX Server FTP, the *junk* subdirectory of the current directory is removed. The same result is achieved by specifying **RMDIR** */u/user1/work/junk*.

```
RMDIR /u/lpn/d.samp
250 RMD command successful.
```

SEND

The **SEND** command is the same as the **PUT** command. Read PUT for details about the **SEND** command.

SEND [*local_path*] [*remote_path*]

Syntax Description

local_path	File name of the file to be copied from the local server.
remote_path	File name of the file at the remote Server into which the file from the local server is copied.

Default

If you omit either path, you are prompted for the file name.

Usage Guidelines

The command name **PUT** can be used in place of **SEND**. There are no other differences when using **PUT**.

The syntax for each path depends on the associated Server FTP.

Example

```
SEND n.d temp
-Data set open with attributes:
Type I N   Stru F   Mode B   Path USER1.N.D
Volser COLPK1   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 3120   Rlse   Bytes/Restart 500000
150 ASCII data connection for temp (138.42.224.15,4115).
-Transfer complete
 6726 bytes sent in 1.58 seconds (4256 bytes/s)   Path USER1.N.D
User COLMBIA   Data bytes sent 6480
Disk tracks read 1
226 ASCII Transfer complete.
```

SITE

The **SITE** (site parameters) command provides the local Server FTP with specific information it requires. This information is essential to file transfers involving that Server FTP, but is not sufficiently universal to have been included specifically in the FTP. Typically, you use a **HELP SITE** Client FTP2 command to find the **SITE** requirements for a specific local Server FTP. Otherwise, review the Server FTP documentation for the **SITE** requirements.

SITE *parameters*

Syntax Description

SITE command parameters are described in the SITE section in Server FTP.

Example

This is an example of a **SITE** command to change the default FTP volume:

```
SITE vol(mvsts2)
```

SNDS

The **SNDS** (resend **SITE** parameters) command directs the Client FTP program to resend the last **SITE** command to the local Server FTP. Your **SITE** command is reissued without your having to retype it. Since most Server FTPs require that new site parameters be provided before each data transfer, the **SNDS** saves time if identical site parameters are to be used repeatedly.

SND

Example

```
SNDS
SITE vol(mvsts2)
SNDS
site vol(mvsts2) <SENT
```

STATUS

The **STATUS** command requests a status response from the local system.

STATUS [*parameter*]

Syntax Description

parameter Specifies the directory to be created.

Defaults

If no parameter is given, the indicated local Server FTP sends status information relative to parameters, such as connection status.

If no data connection is active, the port is zero.

Usage Guidelines

The *parameter* argument is optional.

When the **STATUS** command is issued, the information is transferred over the control connection instead of the data connection.

Example

The Cisco IOS for S/390 Server FTP program implements some additional parameters on the **STATUS** command. Use a **HELP STAT** Client FTP2 command to find additional parameters.

```
STATUS
Connected to: UNIX
--- STATUS ---
-- FTP Parameters --
Remote DT Host, Port 138.42,32,160, 0
Local  DT Host, Port 138.42.224.15, 0
Type I N   Stru F   Mode B   Recall 5   Server is passive
-- END --
-- Control --
User COLMBIA   Acct Accs E0000200   Unit SYSALLDA   Host 138.42.224.15
-- End Control --
-- Path Data --
Volser ICSPK2   Rlse
-- End Path Data --
-- Transfer Information --
Data transfer not in progress   Data bytes sent 6480
Disk tracks read 1   Network bytes sent 6726   Elapsed time 00.00.01
Bytes/Second 4256
-- END --
211 <End of Status>
```

STRUCT

The **STRUCT** (file structure) command provides information on file structure to the remote Server FTP.

STRUCT F | R

Syntax Description

F	File structure is specified by **F**. This is the default if no **STRUCT** command has been used. File structure is used for files with no internal structure, and the file is considered to be a contiguous sequence of data bytes.
R	Record structure is set by **R**. This is for files made up of sequential records. Record structure is accepted for text files (in other words, files with type ASCII or EBCDIC) by all FTP implementations.

Usage Guidelines

One argument is required on the **STRUCT** command. The argument sets the file structure.

Example

```
STRUCT F
```

SUNIQUE

Use the **SUNIQUE (STORE UNIQUE)** command to store transferred files by unique file names on a remote machine. **SUNIQUE** is a toggle command (meaning it is either off or on). When used, the target server automatically ensures that files received in FTP transfer are stored under a unique name.

SUNIQUE

Syntax Description

This command has no arguments or keywords.

Default

Default for **SUNIQUE** is **OFF**.

Usage Guidelines

The target remote FTP server must support the **STOU** (store unique) command.

If **SUNIQUE** is not used, FTP2 will issue the standard **STOR** command. If file names are not unique, files in the target directory could be overwritten.

TYPE

The **TYPE** command tells a Server FTP the data type to use.

TYPE I | **L** *byte_size* | { **A** | **E** [**N** | **T** | **C**] }

Syntax Description

I Indicates image type. The data is sent as a contiguous bit stream that, for transfer, is packed into 8-bit transfer bytes. The receiving site stores the data as contiguous bits.

The receiving storage system might need to pad the file (or each record, in record-structured files) to some convenient boundary. Review the documentation for a Server FTP to find out about padding.

Image type is for the efficient storage and retrieval of files and for transfer of binary data. All FTP implementations are required to support the image type.

L *byte_size* Indicates the local file type and the logical byte size of the file. The byte size value (*byte_size*), representing the logical byte size, is required with the local type. With this type, the data is transferred in logical bytes of the specified size. The logical byte size might differ from the transfer byte size. If the logical and transfer byte sizes differ, the logical bytes are packed contiguously disregarding transfer byte boundaries and are padded at the end if necessary.

When the data reaches the receiving host, it is transformed in a manner dependent on the logical byte size and the particular host. The transformation is invertible; an identical file can be retrieved if the same parameters are used.

The local type is set by **L**. { **A** | **E** [**N** | **T** | **C**] }

A Sets the file type to ASCII. This type is accepted by all FTP implementations and is good for transferring text files, except when both hosts find the EBCDIC type more convenient. In accordance with the *NVT* standard, the CRLF sequence is used at the end of a line of text.

The sender converts the data from an internal character representation to the standard 8-bit NVT ASCII representation (see the Telnet specification in the list of reference documents). The receiver converts the data from this standard form to the receiver's own internal form.

E Sets the files type to EBCDIC. This type performs efficient transfer between hosts that use EBCDIC. Cisco IOS for S/390 Client FTP2 users usually use this type when copying files to their MVS host.

For transmission, data is 8-bit EBCDIC characters. The character code is the only difference between EBCDIC and ASCII types.

End-of-line is rarely used with EBCDIC type to denote structure, but where it is necessary, the NL character is used.

The types ASCII and EBCDIC optionally take a second parameter that indicates what kind of vertical format control, if any, is associated with a file. If a file is to be sent to a host for printing, the receiving host must know how the vertical format control is represented. Therefore, the ASCII and EBCDIC types have a second parameter specifying non-print, Telnet, or carriage control (ASA).

These are the vertical format control specification options:

N Sets non-print format control. This is used when the file does not contain vertical format information. Normally, this format is used with files destined for processing or for storage. Non-print format is accepted by all FTP implementations.

T Sets Telnet format control. This is used when the file contains ASCII/EBCDIC vertical format controls (in other words, CR, LF, NL, VT, FF). The characters CRLF, in exactly this sequence, also denote end-of-line.

C Sets carriage control (ASA) format control. This is used when the file contains ASA (FORTRAN) vertical format control characters.

ASA standard specifies these control characters:

Defaults

A is the default argument for the **TYPE** command.

ASCII is the default file type.

For both ASCII and EBCDIC file types, vertical format control N is the default.

Usage Guidelines

One of the four arguments (I, L *byte_size*, A, or E) is required.

If local type (**L**) is set, the integer byte size argument must also be set.

If ASCII (A) or EBCDIC (E) type is set, one of the three vertical format control arguments, N, T, or C can also can be set.

Example

blank	Move paper up one line.
0	Move paper up two lines.
-	Move paper up three lines.
1	Move paper to top of next page.
+	No movement (in other words, overprint).

Restart Support

If a file transfer is interrupted, it can be restarted. However, restart support requires that mode B be specified.

The restart marker provided by Cisco IOS for S/390 is six bytes long in the format *VTTRBB*. **V** is the volume sequence number, *TTR* is the standard IBM OS disk block address, and *BB* is a byte offset within the block.

Using Restart

Use the **SITE REST**(*XXXXX*) Client FTP2 command to tell a Cisco IOS for S/390 FTP Server how often to send a restart marker. Send a restart marker after the sending of the record that exceeds or equals *XXXXX* number of bytes (varying between 1 and 500,000). Send the **SITE** command only to the **RETR** side of a data transfer. Other FTP Servers may initiate sending restart markers in a different way from Cisco IOS for S/390.

A 110 message is written once per output block if a restart marker is sent somewhere in the data written for the block.

For example, suppose during a data transfer this restart mark message is sent:

```
110 MARK 0100030212C0 = 010003020FA0
```

You can restart the transfer at this point by sending the first number to the **RETR** side and the second number to the **STOR**. If you receive this restart mark message during an aborted data transfer, you can restart the transfer at these disk locations with these Client FTP2 **REST** commands:

Send this command to the RETR side:

REST 0100030212C0

Send this command to the STOR side:

REST 010003020FA0

Cisco IOS for S/390 supports restart markers (set at default value of every 32767 data bytes) if these conditions exist:

- TYPE I
- MODE B

Client FTP2 File Transfer Examples

This section provides some examples of file transfers.

Example Notes

These notes clarify the Client FTP2 examples included in this section:

- Issue **FTP2** when under TSO to enter Client FTP2.
- Text can be entered in uppercase or lowercase. Some host systems support a mixture of lowercase and uppercase letters, while other host systems use uppercase for most functions. All commands entered are translated to uppercase before being sent to the servers. The data associated with a command is sent to its appropriate FTP server without case translation. The Cisco IOS for S/390 FTP server translates user IDs, passwords, data set names, and similar items to uppercase before the commands associated with them are executed.
- An **OPEN** command attempts to connect to a remote host. FTP2 attempts to log the user on to the remote host once the connection is made.

The examples are shown in the following sections.

PUT Example

In the following example FTP2 session, the **PUT** command transfers a file from local MVS host to remote host unix.

```
Setting local directory to the TSO profile prefix
250 "USER1." is current prefix
Cisco IOS for S/390 Release 2.0 Client FTP2 - Enter command or'?'
open unix
220 unix FTP server (SunOS 4.1) ready.
Enter name (UNIX:user1sys1): user_name
331 Enter PASS command
Password: passwd
230 User user1 loggged in.
put cntl(iefbr14) jclbr14
-Data set open with attributes:
Type A N  Tabs 8  Stru F  Mode S  Path USER1.CNTL(IEFBR14)
Volser COLPK1  Unit SYSALLDA  Dsorg PO  Recfm FB  Lrecl 80  Blksize 3120  Rlse
150 ASCII data connection for jclbr14 (138.42.224.15,4120).
-Transfer complete
820 bytes sent in 0.26 seconds (3153 bytes/s)
Path USER1.CNTL(IEFBR14)
User USER1 Data bytes sent 800
Disk tracks read 1
226 ASCII Transfer complete.
end
```

- FTP2 makes a connection to the MVS local host. The connection is established automatically unless the NOA invocation option was specified.

  User USER1SYS1 is logged on to the local host using information gathered in the TSO address space. FTP2 discovered that user USER1SYS1 has a different TSO profile prefix than his user ID. FTP2 set the local working directory to his current TSO profile prefix of USER1.

- The **open unix** command establishes a connection to the remote UNIX host. FTP2 automatically prompts for a user ID, password, and optionally, an account.

- At the "Enter name..." entry, FTP2 prompts you for the correct user ID on UNIX.

  User USER1SYS1 can press **Enter** and get a default user ID of user1sys1, or can override the user ID and enter a different user ID.

  A user ID of user1 was entered. A password is prompted for and entered in a non-display field. The user ID and password are valid and user user1 logs on to the remote host.

- The **put cntl(iefbr14) jclbr14** command tells FTP to transfer the file USER1.CNTL(IEFBR14) from the local host (MVS) and create or overwrite file jclbr14 on the remote host, unix.

  The 226 message indicates a successful transfer occurred. Any 400 or 500 level message indicates a data transfer failure.

- The **end** command ends the FTP2 session.

GET Example

In the following example FTP2 session, the **GET** command transfers a file from remote host unix to the local MVS host.

```
Cisco IOS for S/390 Rn Client FTP2 - Enter command or '?'
open unix
220 unix FTP server (SunOS 4.1) ready.
Enter name (UNIX:user1): user_name
331 Enter PASS command
Password: passwd
230 User user1 logged in.
get jclbr14 cntl(newbr14)
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.CNTL(NEWBR14)
Volser COLPK1   Unit SYSALLDA   Dsorg PO   Recfm FB   Lrecl 80
Blksize 3120   Space 31 15 Tracks Rlse
150 ASCII data connection for jclbr14 (138.42.224.15,4121) (810 bytes).
-Transfer complete
 820 bytes received in 10.1 seconds (80 bytes/s)
Path USER1.CNTL(NEWBR14)   User USER1   Data bytes received 800
Disk tracks written 1
226 ASCII Transfer complete.
end
```

- The **open unix** command establishes a connection to the remote host **unix**.

  A connection is made to the MVS local host. **user1** is logged on to the local host using information gathered in the TSO address space.

  After establishing a connection, FTP2 automatically prompts for a user ID, password, and, optionally, an account.

- At the "Enter name..." entry, FTP2 prompts user1 for the correct user ID on unix. At this prompt, user1 can press **Enter** and get a default user ID of user1, or can override the user ID and enter a different user ID.

  In this example, **user1** just hit the enter key, so the default user ID is used. A password is prompted for and entered in a non-display field. The user ID and password are valid and **user1** logs on to the remote host.

- The command **get jclbr14 cntl(jclbr14)** tells FTP to transfer the file jclbr14 on remote host unix to file USER1.CNTL(NEWBR14) on the local MVS host.

  The 226 message indicates a successful transfer. Any 400 or 500 level message indicates a data transfer failure.

- The **End** command ends the FTP2 session.

FTP2 Restart Marker Example

When transferring large amounts of data across many channels, the transfer sometimes does not complete successfully. The following example FTP2 session shows how to use the **rest** (restart) command to imbed marks to tell Server FTP where to restart a file transfer. Read REST.

```
Cisco IOS for S/390 Rn - Usr FTP2 - Enter command or '?'
open mvs
220 MVS.HQ.COMPANY.COM -- FTP2 Server, Enter command or HELP
Enter name (MVS:columbia): user1
331 Enter PASS command
Password: passwd
230 LOGGED IN - HOST 138.42.128.13 USER user1
binary
mode b
site rest(100000)
remsite space(3 1) cyl recfm(vb) blk(6650) lrecl(133)
put 'scm.p016572.T01TCP' psr16572.job
-Dataset open with attributes:
Type I N   Stru F   Mode B   Path SCM.P016572.T01TCP
Volser HAGCAT   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 133
Blksize 6650 Rlse Bytes/Restart 100000
150-Dataset open with attributes:
Type I N   Stru F   Mode B   Recall 5   Path USER1.PSR16572.JOB
Volser HAGCAT   Unit SYSALLDA   Dsort PS   Recfm VBV   Lrecl 133
Blksize 6650 Space 3 1 Cyl Rlse
-Transfer complete
 1656466 bytes sent in 13.5 seconds (122338 bytes/s)
Path SCM.P016472.T01TCP   User COLMBIA   Data bytes sent 1619142
Disk tracks read 41   Restart markers sent 16
110    MARK    000204010a      = 00020414F9
110    MARK    0005010214      = 0005021075
110    MARK    000704031E      = 0007060BF1
110    MARK    000A010428      = 000A040771
110    MARK    000C040532      = 000D0202ED
110    MARK    000F01063C      = 000F0517DE
110    MARK    0011040746      = 001203135E
110    MARK    0014010850      = 0015010EDA
110    MARK    001604095A      = 0017050A56
110    MARK    0019010A64      = 001A0305D6
110    MARK    001B040B6E      = 001D010152
110    MARK    001D010C78      = 001F041643
110    MARK    0020040D82      = 00220211C3
110    MARK    0020040D82      = 00220211C3
110    MARK    0023010E8C      = 0024060D3F
110    MARK    0025040F96      = 00270408BB
110    MARK    00280110A0      = 002A02043B
226-Transfer complete
1656466 bytes received in 13.6 seconds (120998 bytes/s)
Path USER1.PSR16572.JOB  User USER1   Data bytes received 1619142
Disk tracks written 43              Records folded 12551
Restart markers received 16
```

- FTP2 starts up and automatically connects and signs on to the local host. Then the user connects and signs on to the remote host mvs.
- The “binary” entry sets the data type to binary.
- The “mode b” entry invokes block mode, which tells the FTP servers that restart markers are being used.
- The “site rest(100000)” entry tells the local FTP server how often to place restart markers into the data.

- The **remsite** command issued to the remote site allocates a data set with the correct attributes.
- The file SCM.PO16572.MYFILE is transferred from the local MVS host to host mvs as USER1PSR16572.JOB. All the data was successfully transferred.

  If the data transfer fails you can restart the data transfer using any of the "110 MARK ..." restart markers.

  Read REST for more details.

Transfer to an MVS Internal Reader

The following example FTP2 session shows an FTP2 file transfer from a file on host unix to an MVS internal reader on the local MVS host.

```
Cisco IOS for S/390Rn Client FTP2 - Enter command or '?'
open unix
220 unix FTP server (SunOS 4.1) ready.
Enter name (UNIX:user1): user_name
331 Enter PASS command
Password: passwd
230 User user1 logged in.
site submit
get jclbr14 anyname
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Intrdr   Recfm FB   Lrecl 80   Blksize 20000
150 ASCII data connection for jclbr14   (138.42.224.15,4122) (810 bytes).
-Transfer complete
 820 bytes received in 4.17 seconds (196 bytes/s)   User USER1
Data bytes received 800
226 ASCII Transfer complete.
end
```

- The **open unix** command establishes a connection to the remote host **unix**. The user logs on to the remote host.
- The command **site submit** is a command to the local MVS host directing the next data transfer to the MVS internal reader for execution. **SITE** commands are relevant to an MVS host where Cisco IOS for S/390 is running.
- The command **get jclbr14 anyname** commands FTP2 to transfer file jclbr14 from remote host unix. Due to the previous **SITE** command, the file is transferred to an MVS internal reader on the local MVS host. The file name (anyname) for the local host is ignored because no data set is being created or updated.
- The **end** command ends the FTP2 session.

Managing Directories on UNIX-based Systems

The following example FTP2 session shows the directory manipulation commands for the remote host.

```
Cisco IOS for S/390 Rn Client FTP2 - Enter command or '?'
open unix
220 unix FTP2 server (SunOS 4.1) ready.
Enter name (UNIX:user1): user_name
331 Enter PASS command
Password: passwd
230 User user1 logged in.
pwd
257 "/home/unix/user1" is current directory.
mkdir tempdir
A:257 MKD command successful.
pwd
257 "/home/unix/user1" is current directory.
cwd tempdir
250 CWD command successful.
pwd
257 "/home/unix/user1/tempdir" is current directory.
put cntl(iefbr14) jclbr14
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Path USER1.CNTL(IEFBR14)
Volser COLPK1   Unit SYSALLDA   Dsorg PO   Recfm FB   Lrecl  80
Blksize 3120   Rlse
150 ASCII data connection for jclbr14 (138.42.224.15,4123).
-Transfer complete
 820 bytes sent in 0.54 seconds (1518 bytes/s) Path USER1.CNTL(IEFBR14)
User USER1   Data bytes sent 800
Disk tracks read 1
226 ASCII Transfer complete.
dir
-Data set open with attributes:
Type A N   Tabs 8   Stru F   Mode S   Recall 5
Path USER1.FTP.TMP.T2334662
Volser ICSUSR   Unit SYSALLDA   Dsorg PS   Recfm FB   Lrecl 80
Blksize 6160   Space
5 3 Tracks Rlse
150 ASCII data connection for /bin/ls (138.42.224.15,4124) (0 bytes).
226 ASCII Transfer complete.
-Transfer complete
 72 bytes received in 9.29 seconds (7 bytse/s)
Path USER1.FTP.TMP.T2334662   User COLMBIA   Data bytes received 68
Disk tracks written 1   Records padded 2
total 1
-rw-rw-rw-  1 user1   dvlp   810 Jan 26 10:19 jclbr14
IDC0550I ENTRY (A) USER1.FTP.TMP.T2334662 DELETED
dele jclbr14
250 DELE command successful.
cdup
250 CWD command successful.
pwd
257 "/home/unix/user1" is current directory.
rmd tempdir
250 RMD command successful.
pwd
257 "/home/unix/user1" is current directory.
end
```

- User1 connects and logs on to the local host and remote host unix.
- The **pwd** command asks the server to print the current directory on the remote host.

- The command **mkdir tempdir** asks the remote ftp host server to create a directory called tempdir.
- By creating a directory in the third section of this example, you have not changed your current working directory on the remote host.

  The command **cwd tempdir** changes the remote host directory from /home/unix/user1 to /home/unix/user1/tempdir. The **pwd** command confirms the directory.
- The **put cntl(iefbr14) jclbr14** command copies a file from the local MVS host to the remote host as file jclbr14.
- The **dir** command finds all the current members on the current remote host directory. The only file in the directory is jclbr14.
- The **dele jclbr14** command removes the file from the remote host.
- The command **cdup** changes the remote host directory from its current directory /home/unix/user1/tempdir to its parent directory home/unix/user1. The **pwd** command reflects this change.
- The command **rmd tempdir** asks the remote host server to remove directory /home/unix/user1/tempdir.
- The **end** command terminates the FTP2 session.

Transferring and Using a File in a Single JCL Job

Transferring a file in one job step and using that file in another job step can run into a file allocation problem, wherein the file transfer fails in the FTP2 job step. This is usually because the file has been allocated to the batch job and the Cisco IOS for S/390 base product cannot allocate the file for a data transfer. The FTP2 and FTP client programs do not perform file transfers in their own address space; the FTP2 and FTP clients direct the Cisco IOS for S/390 base product to perform file transfers.

As a workaround, perform an IDCAMS ALTER NEWNAME of the file between the file transfer step and the use of the file.

Examples

The following sample JCL does a file transfer, performs an IDCAMS ALTER NEWNAME, and uses the file in the final job step (the data set MVS.P25206.DATA is created in the first job step, renamed to the file MVS.NEWNAME.DATA in the second step, and the file name MVS.NEWNAME.DATA is used in the last step):

```
//MVSA JOB (TSO00...99),'FTP2 BATCH',MSGCLASS=X,NOTIFY=MVS,CLASS=A
//*
//*  JOB TO TRANSFER A FILE AND THEN USE IT IN A LATER JOBSTEP
//*
//*  STEP TO DO THE FTP TRANSFER
//*
//FTP1        EXEC PGM=IKJEFT01,REGION=4000K
//SYSTSIN     DD *,DCB=BLKSIZE=80
    FTP2 / APP=ACCES  TEST NETRC
//STEPLIB     DD DCB=BLKSIZE=32000,
//               DISP=SHR,DSN=T01TCP.LINK
//SYSTSPRT    DD SYSOUT=X
//SYSPRINT    DD SYSOUT=*,DCB=BLKSIZE=133
//SYSPUT      DD SYSOUT=*,DCB=BLKSIZE=133
//SYSVLT      DD SYSOUT=*,DCB=BLKSIZE=133
//NETRC       DD DISP=SHR,DSN=MVS.FTP.NETRC
//SYSGET      DD *,DCB=BLKSIZE=80
open unix
get temp 'mvs.p25206.data'
END
/*
//*
//*   STEP TO PERFORM AN IDCAMS ALTER NEWNAME
//*
//STEP2       EXEC PGM=IDCAMS
//SYSPRINT    DD SYSOUT=*
//SYSIN       DD *
 ALTER 'MVS.P25206.DATA'                NEWNAME('MVS.NEWNAME.DATA') -
  CAT(CATALOG.TSO.VESA001)
//*
//*   THIS STEP PRINTS THE FILE ON THE SYSUT1 DD CARD
//*
//STEP3       EXEC PGM=IEBGENER
//SYSUT1      DD DISP=SHR,DSN=MVS.NEWNAME.DATA
//SYSUT2      DD SYSOUT=X,COPIES=1,DCB=*.SYSUT1
//SYSPRINT    DD   SYSOUT=*
//SYSIN       DD   DUMMY
```

The following sample JCL uses a file, performs an IDCAMS ALTER NEWNAME, and does a file transfer in the final job step (the file MVS.OLDNAME.DATA is used in the first jobstep, renamed to the file MVS.TRANSFER.DATA in the second jobstep, and the file MVS.TRANSFER.DATA is transferred in the last jobstep):

```
//MVSB JOB (TSO00,,,99),'FTP2 BATCH',MSGCLASS=X,NOTIFY=MVS,CLASS=A
//*
//* JOB TO TRANSFER A FILE AND USE IT IN A LATER JOBSTEP
//*
//*   THIS STEP PRINTS THE FILE ON THE SYSUT1 DD CARD
//*
//STEP1     EXEC PGM=IEBGENER
//SYSUT1    DD DISP=SHR,DSN=MVS.OLDNAME.DATA
//SYSUT2    DD SYSOUT=X,COPIES=1,DCB=*.SYSUT1
//SYSPRINT  DD    SYSOUT=*
//SYSIN     DD    DUMMY
//*
//*   STEP TO PERFORM AN IDCAMS ALTER NEWNAME
//*
//STEP2     EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
 ALTER 'MVS.OLDNAME.DATA'                NEWNAME('MVS.TRANSFER.DATA') -
  CAT(CATALOG.TSO.VESA001)
//*
//*  STEP TO DO THE FTP TRANSFER
//*
//FTP3      EXEC PGM=IKJEFT01,REGION=4000K
//SYSTSIN   DD *,DCB=BLKSIZE=80
FTP2 / APP=ACCES  TEST NETRC
//STEPLIB   DD DCB=BLKSIZE=32000,
//          DISP=SHR,DSN=T01TCP.LOAD
//SYSTSPRT  DD SYSOUT=X
//SYSPRINT  DD SYSOUT=*,DCB=BLKSIZE=133
//SYSPUT    DD SYSOUT=*,DCB=BLKSIZE=133
//SYSVLT    DD SYSOUT=*,DCB=BLKSIZE=133
//NETRC     DD DISP=SHR,DSN=MVS.FTP.NETRC
//SYSGET    DD *,DCB=BLKSIZE=80
open unix
put 'mvs.transfer.data' temp
END
/*
```