## Contents

**FTP** ............................................. 1  
What’s new for V5R1 .................................. 1  
Print this topic ..................................... 2  
Managing FTP on iSeries 400 .......................... 2  
  Setting up FTP servers for graphical FTP clients, Web browsers, and other Web tools  2  
Anonymous FTP ......................................... 3  
  Preparing for Anonymous FTP .......................... 4  
  Exit programs for Anonymous FTP ........................ 5  
  Creating an OS/400 user profile: ANONYMOUS. ........................ 6  
  Creating a public library or directory .......................... 6  
  Installing and registering exit programs .................. 6  
Implementing FTP security ................................ 7  
  Monitoring incoming FTP users .......................... 7  
  Managing user access to FTP functions ................... 7  
Securing FTP with exit programs ............................ 8  
  Validation exit point for TCP/IP application servers ........................ 8  
  Logon exit point for TCP/IP application servers .................. 12  
  Removing exit programs .................................. 28  
Improving FTP server performance with configurable subsystem support ............. 28  
Working with the FTP client .............................. 28  
  Starting and ending a client session .................... 29  
  Starting the FTP client session ......................... 29  
  Ending the FTP client session ......................... 29  
File systems and naming conventions .......................... 29  
  Transferring files with FTP ............................ 29  
  File transfer types ..................................... 30  
FTP as batch job ........................................ 30  
  Batch FTP: A Complex Example .......................... 32  
FTP client subcommands .................................... 37  
  ACCT (Send Account Information) ......................... 41  
  APPEND (Append a Local File Member to a Remote File) .................... 41  
  ASCII (Change File Type to ASCII) ........................ 42  
  BINARY (Set Transfer Type to Image) ....................... 42  
  CD (Change Working Directory or Library) .................. 43  
  CLOSE (End an FTP Session with the Remote System) .................. 43  
  DEBUG (Create Client Trace and Control Display of Server Subcommands Sent to Remote System) .......................... 43  
  DEBUG (Change Client Time-Out Limit Values) .................. 44  
  DELETE (Delete a File on a Remote System) .................. 45  
  DIR (List Directory Entries, Libraries, or Files) .................. 45  
  EBCDIC (Change File Type to EBCDIC) ...................... 46  
  GET (Copy a File from a Remote System to the Local System) .................. 46  
  HELP (Getting Help for FTP Subcommands) ................... 47  
  LCD (Change Working Library or Directory on Local System) .................. 48  
  LOCSITE (Specify Local Site Information) .................. 48  
  LOCSSTAT (Display Local Status Information) .................. 49  
  LS (List Remote File Names) ............................ 50  
  LTYPE (Local Type) ..................................... 50  
  MDELETE (Delete Multiple Files on a Remote System) .................. 51  
  MGET (Copy Multiple Files from a Remote System to the Local System) .................. 51  
  MKDIR (Make Directory) ............................... 52  
  MODE (Specify Transmission Mode of Data) ................... 52  
  MPUT (Send Multiple File Members from the Local System to a Remote System) .................. 53
FTP

You can set up your iSeries 400 system to send, receive, and share files across networks by using the file transfer protocol (FTP). You can also rename, add, and delete files. Before you set up your system to transfer files, you must have TCP/IP configured and started on your system.

For more information about using FTP on your iSeries 400 system, see the following:

What’s new for V5R1
See what functions are new to the iSeries for FTP in V5R1.

Print this topic
Download or print the FTP documentation in a PDF format.

Managing FTP on iSeries 400
Learn how to set up FTP servers for graphical FTP clients, Web browsers, and Web tools, allow anonymous FTP, and secure your FTP server.

Working with the FTP client
Learn how to start and end client sessions, transfer and receive files, and use FTP client subcommands.

Working with the FTP server
Learn how to start and stop the FTP server, about batch FTP, and FTP server subcommands.

Troubleshooting FTP
Get information on troubleshooting problems with the FTP server or FTP client.

Tip: A number of FTP procedures are available in the online help from Operations Navigator.

What’s new for V5R1

FTP on the iSeries 400 includes the following enhancements for V5R1:

- Manage user access to FTP functions
  New limit access by function support allows a system administrator to manage which FTP subcommands one can use.
- SSL to secure FTP connections
  Secure sockets layer support allows the user to to eliminate the exposure of sending passwords and data “in the clear” on the network when using the OS/400 FTP server with an FTP client that also supports SSL
- TCPL0300 exit point format
  This new exit point supports OS/400 enhanced password.
- Improving FTP server performance with configurable subsystem support
  The addition of configurable subsystem support allows the system programmer or administrator to define an OS/400 subsystem for FTP server jobs to enhance performance and manageability.
- Improved file transfer performance
  Faster data transfer rates will enable you to send and receive files through FTP more quickly.
- Ability to connect to a nonstandard port
  When you start an FTP connection, you can now specify which port to connect with on the remote system. This will allow you to transfer files with FTP servers that do not use the standard FTP port.
Managing FTP on iSeries 400

You can set up your iSeries 400 to send, receive, and share files across networks by using file transfer protocol (FTP). FTP consists of two parts: the FTP client and the FTP server. You interact with the FTP client. The FTP client interacts with the FTP server. You do not normally interact directly with the FTP server.

- **Setting up FTP servers for graphical FTP clients, Web browsers, and other Web tools**
  Setting up your FTP server to work with graphical clients allows remote users to interact with your server using familiar point-and-click methods.

- **Anonymous FTP**
  Anonymous FTP enables remote users to log on to your FTP server and download files even though they do not have a userid and password.

- **Implementing FTP security**
  Secure your FTP server by monitoring incoming FTP users, preventing hacker access, using Secure Sockets Layer (SSL) protection, and with FTP exit point programs.

- **Improving FTP performance with configurable subsystem support**
  You can change the subsystem that the FTP server runs in to help improve performance of the FTP server.

### Setting up FTP servers for graphical FTP clients, Web browsers, and other Web tools

The FTP server on the iSeries 400 system supports graphical FTP clients, Web browsers, and Web development tools.

Most graphical FTP clients use UNIX as their list format and path file as their file name format. Follow these instructions to set the FTP server properties to use the supported formats.

1. In Operations Navigator, expand your system.
2. Expand **Network**.
3. Expand **Servers**.
4. Click **TCP/IP**.
5. Right-click **FTP** and select **Properties**.
6. The **Properties** dialog appears. Click on the **Initial Formats** tab.
7. Enable Path as the File Naming Format.
8. Enable UNIX list format as the File List Format.
**Note:** These two formats become your defaults for your FTP server. You can set the LISTFMT and NAMEFMT on the server with an FTP Server Logon Exit program using the TCPL0200 format or TCPL0300 format for the FTP Server Logon Exit Point.

You can also change the list format after an FTP session is in progress with options for the FTP server SITE (Send Information Used by a Server System) subcommand. To get a list of directory entries, library contents, or files in a file group, use the LIST (File List) FTP subcommand. To sort file names in a set of files, file group, directory, or library, use the NLST (Name List) FTP subcommand.

See also:
- LIST Information in iSeries 400 format
- LIST information in UNIX-style format

### Anonymous FTP

Anonymous FTP enables unprotected access (no password required) to selected information on a remote system. The remote site determines what information is made available for general access. Such information is considered to be publicly accessible and can be read by anyone. It is the responsibility of the person who owns the information and the system to assure that only appropriate information is made available.

To access this information, a user logs on to the hosts using the user ID ANONYMOUS. The user ANONYMOUS has limited access rights to the files on the server and has some operating restrictions. Typically, the only operations allowed include the following:
- Logging on using FTP
- Listing the contents of a limited set of directories
- Retrieving files from these directories.

Usually, anonymous users are not allowed to transfer files to the FTP server system. Some systems do provide an incoming directory for anonymous users to send data to. Traditionally, the special anonymous user account accepts a string as a password, although it is common to use either the password 'guest' or one's electronic mail (e-mail) address. Some archive sites explicitly ask for the user’s e-mail address and will not allow logon with the guest password. Providing an e-mail address is a courtesy that allows the archive site operators to get some idea of who is using their services.

### Anonymous FTP on the iSeries 400

The basic File Transfer Protocol (FTP) server does not support anonymous FTP. To set up anonymous FTP on your iSeries 400 system, you need to provide exit programs for the two FTP Server exit points (server logon exit point and request validation exit point).

You may want to provide anonymous FTP because it is a convenient and often necessary service. However, the use of anonymous FTP raises system security issues for your iSeries 400. See Securing FTP with exit programs "Securing FTP with exit programs" on page 8 for more information about protecting your system.

**Follow these steps to set up Anonymous FTP:**

1. **Preparing for anonymous FTP** "Preparing for Anonymous FTP" on page 4: Check to see if you are ready for Anonymous FTP.
2. **Writing two FTP exit programs** "Exit programs for Anonymous FTP" on page 5: Write two exit programs for Anonymous FTP support. This section provides many examples of exit programs as well as useful tips and techniques.
3. **Creating an OS/400 User Profile: ANONYMOUS** "Creating an OS/400 user profile: ANONYMOUS" on page 6: Prevent anyone from signing on to your iSeries 400 directly with a user ID of ANONYMOUS.
4. Creating a Public Library or Directory. Create, load, and set your public libraries or directories.

5. Installing and registering exit programs. Create a library to contain your exit programs and their log files, compile the programs, and register them for use by the FTP server.

Preparing for Anonymous FTP
The following information will help you prepare to set up Anonymous FTP.

Skill Requirements
To set up Anonymous FTP, you will need the following skills:

- You should be familiar with your iSeries 400 command line interface and commands with multiple parameters and keywords.
- You must be able to create libraries, members, and source physical files on your iSeries 400 (you should have at least *SECOFR authority).
- You must be able to assign authorities to libraries, files, members, and programs.
- You must be comfortable writing, modifying, compiling, and testing programs on your iSeries 400.

Example programs
Example programs are available to help you set up anonymous FTP on your server. You can use these samples as a starting point to build your own programs. By copying portions of the code from the samples, you can add them to programs that you write yourself. It is recommended that you run the sample programs on a system other than your production system.

Examples programs include the following:

- Example: FTP Server Logon exit program in C code
- Example: FTP Server Request Validation exit program in iSeries 400 CL code
- Example: FTP Server Logon exit program in ILE RPG code

Safeguards to consider
It is not a good idea to use your iSeries 400 system as an FTP server on the Internet without these safeguards:

1. Defining your anonymous FTP server site policy
2. Use a firewall between your iSeries 400 system and the Internet.
3. Use a non-production iSeries 400 system for your FTP server system.
4. Do not attach the FTP server to the rest of your company’s LANs or WANs.
5. Use FTP exit programs to secure access to the FTP server.
6. Test FTP exit programs to ensure that they do not contain security loopholes.
7. Do not allow anonymous FTP users to have read and write access to the same directory. This permits the anonymous user to be untraceable on the Internet.

What to do next:
- Writing two FTP exit programs. Write two exit programs for Anonymous FTP support.
Exit programs for Anonymous FTP
To set up Anonymous FTP support, you need to write two exit programs: FTP Server Logon exit program and FTP Server Request Validation exit program. The FTP Server Logon exit program enables the ANONYMOUS user ID and forces the ANONYMOUS user to the public library or directory. The FTP Server Request Validation exit program restricts the commands, files, and directories or libraries that the ANONYMOUS user can use.

Exit points and exit point formats
The FTP server communicates with each exit program through a specific exit point. Parameters are passed between the server and the exit program. The format of the exchanged information is specified by an exit point format.

Note:
An exit point may have more than one format, but an exit program can only be registered for one of the exit point formats.

iSeries 400 exit points and their formats for each exit program:

<table>
<thead>
<tr>
<th>Program</th>
<th>FTP Server Logon exit program</th>
<th>FTP Server Request Validation exit program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exit Point</td>
<td>QIBM_QTMF_SVR_LOGON</td>
<td>QIBM_QTMF_SERVER_REQ</td>
</tr>
<tr>
<td>Format</td>
<td>TCPL0100, TCPL0200, or TCPL0300. Examine each of these formats, then choose the one most appropriate for your system.</td>
<td>VLRQ0100</td>
</tr>
</tbody>
</table>

Also see:
- Logon exit point for TCP/IP application servers
- Validation exit point for TCP/IP application servers

Exit programs for the FTP server exit points:
- The FTP Server Logon exit point program
- The FTP Server Request Validation exit point program

Exit program code examples
Note: These examples are for illustration purposes only. They do not contain enough features to run on a production machine as is. Feel free to use them as a starting point, or to use sections of code as you write your own programs.
- Example: FTP Server Logon exit program in CL code
- Example: FTP Server Logon exit program in C code
- Example: FTP Server Logon exit program in ILE RPG code
- Example: FTP Server Request Validation exit program in ILE RPG code
What to do next:

- Creating an OS/400 user profile: ANONYMOUS

Prevent anyone from signing on to your iSeries 400 directly with a user ID of ANONYMOUS.

Creating an OS/400 user profile: ANONYMOUS

To prevent anyone from directly signing on to your iSeries 400 with the user profile ANONYMOUS, it is strongly recommended that you create a user profile of ANONYMOUS and assign it a password of *NONE. You can create this profile through Operations Navigator.

1. In Operations Navigator, expand the system that you are using as your central system.
2. Expand Users and Groups. Use the help for additional assistance.

What to do next:

- Creating a Public Library or Directory

Create, load, and set your public libraries or directories.

Creating a public library or directory

1. Create the public libraries or directories that will contain files accessible through anonymous FTP.
2. Load your public libraries or directories with the public access files.
3. Set the public libraries or directories and file authorities to PUBLIC *USE.

What to do next:

- Installing and registering exit programs

Create a library to contain your exit programs and their log files.

Installing and registering exit programs

Install the exit program

1. Create a library to contain your exit programs and their log files.
2. Compile your exit programs in this library.
3. Grant PUBLIC *EXCLUDE authority to the library, program, and file objects.
   The FTP server application adopts authority when necessary to resolve and call the exit program.

Register the exit program

1. Enter WRKREGINF at an iSeries 400 command line.
2. Page down to an FTP Server Logon exit point:
   - QIBM_QTMF_SVR_LOGON TCPL0100
   - QIBM_QTMF_SVR_LOGON TCPL0200
   - QIBM_QTMF_SVR_LOGON TCPL0300
   - QIBM_QTMF_SERVER_REQ VLRQ0100
3. Enter 8 in the Opt field to the left of the exit point entry and press Enter.
4. At the Work with Exit Programs display, enter a 1(add).
5. Enter the name of the exit program in the Exit Program field.
6. Enter the name of the library that contains the exit program in the Library field.
7. Press Enter.
8. End and restart the FTP server to ensure that all FTP server instances use the exit programs.
9. Test your exit programs thoroughly.

Note:

Exit programs take effect as soon as the FTP server requests a new FTP session. Sessions that are already running are not affected.

Related topic:
Implementing FTP security

Attention to FTP Security is necessary to ensure that vital business data stored on your FTP server is not compromised.

You can find information about ways to protect your server in the following topics:

- **Monitoring incoming FTP users**
  Monitor who is logging in to your FTP server.

- **Managing user access to FTP functions**
  Learn how to limit access to FTP server functions.

- **Using SSL to secure FTP**
  With a few simple steps, you can choose the level of security for your FTP server by allowing only SSL connections, only non-SSL connections, or both SSL and non-SSL connections.

- **Securing FTP with exit point programs**
  Find information on how you can use FTP exit programs to increase or change FTP access to your system.

**Monitoring incoming FTP users**

If you want to monitor for incoming FTP users, enter:

```
WRKSBSJOB SBS(QSYSWRK)
```

The format for the names of these jobs is `QFTPnnnnn`. The `nnnnn` is a randomly-generated number.

You can start FTP server jobs by running one of these commands:

- Start TCP (STRTCP) command with the Autostart parameter set to *YES
- Start TCP server (STRTCPSVR) command with a Server parameter of *FTP or *ALL

**Managing user access to FTP functions**

Beginning with V5R1, you can use Operations Navigator to limit user access to FTP server and client functions. You use Application Administration to grant and deny access to functions for individual users or for groups of users. Alternatively, you can manage access to FTP functions by writing FTP exit programs for the FTP Request Validation Exit Points.

To manage user access to functions through Operations Navigator, complete the following steps:

1. In Operations Navigator, right-click your iSeries 400 system, then select **Application Administration**.
2. Select **Host Applications**.
3. Expand **AS/400 TCP/IP Utilities**.
4. Expand **AS/400 File Transfer Protocol**.
5. Expand **FTP Client Operations** or **FTP Server Operations**.
6. Select the function that you want to allow or deny access to.
7. Click **Customize**.
8. Use the **Customize Usage** dialog to modify the list of users and groups that are allowed or denied access to the function.
9. Click **OK** in the **Customize Usage** dialog.
10. Click **OK**.

Alternatively, you can manage the access that a specific user or group has to registered FTP functions through Operations Navigator’s Users and Groups management tool. To do this, follow these steps:

1. Expand your iSeries 400 system.
2. Expand **Users and Groups**.
3. Select **All Users** or **Groups**.
4. Right-click on a user or group, then select **Properties**.
5. Click **Capabilities**.
6. Click **Applications**.

   From here, you can change the user or group's settings for the listed function. You can also change the settings for all functions in a hierarchy grouping by changing the settings of the “parent” function.

For more information on securing your iSeries FTP server, see the Implementing FTP security topic.

**Securing FTP with exit programs**

FTP provides a security level based on the OS/400 object security. This means that remote users cannot logon to your iSeries 400 FTP server unless they have a valid user profile and password.

You can provide additional security by adding exit programs to the FTP Server and Client exit points to further restrict FTP access to your system. For example, you can restrict FTP logon capability, as well as access to libraries, objects, and the use of commands.

You can write an FTP Server Request Validation exit program in order to restrict the CL commands and FTP subcommands that users may access. For instructions, see the Validation exit point for TCP/IP application servers subtopic.

Beginning with V5R1, you can limit access to FTP subcommands through Operations Navigator. To do this, see the Managing user access to FTP functions subtopic. Depending on your situation, you may consider using this method as an alternative to writing exit programs for the the FTP Server Request Validation and FTP Client Request Validation exit points. You can write an FTP Client Request Validation exit program for the client exit point format. This controls which FTP client functions a user may perform.

You can control the authentication of users to a TCP/IP application server with the Logon exit point for TCP/IP application servers subtopic. To allow the exit programs to work properly, you must installing and register your exit point programs. If your programs are no longer needed, you must properly removing the exit point programs to prevent their future functioning.

**Validation exit point for TCP/IP application servers:**

**What your program should include:**

- Exception handling
- Debugging
- Logging

**Write your FTP Server Request Validation exit program in order to:**

1. Restrict anonymous FTP users to FTP retrieve subcommands only, and never allow anonymous users to execute CL commands.

**Allowed and rejected commands**

The FTP request validation exit program gives you control over whether to accept or reject an operation. Decisions made by exit programs are in addition to any validation that is performed by the FTP server application. The FTP server calls the FTP server request validation exit program each time it processes one of these requests:

1. Directory/library creation
2. Directory/library deletion
3. Setting current directory
4. Listing file names
5. File deletion
6. Sending a file
7. Receiving a file
8. Renaming a file
9. Executing a CL command on the FTP server

You may want to set value -1 of parameter 8 (Allow operation) in the VRLQ0100 exit point format to always and unconditionally reject a command.

Is there an exit program time-out feature?

There is no time-out for FTP exit programs. If the exit program has an error or exception that it cannot handle, the FTP server will abort the session.

VRLQ0100 exit point format: The exit point for FTP Server Application Request Validation is:

QIBM_QTMF_SERVER_REQ

The exit point for FTP Client Application Request Validation is:

QIBM_QTMF_CLIENT_REQ

The interface that controls the parameter format for the exit point is:

VLRQ0100

The table below shows the parameters and parameter format for the VLRQ0100 interface.

### Required Parameter Format for the VLRQ0100 exit point interface

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Input or Output</th>
<th>Type and length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application identifier</td>
<td>Input</td>
<td>Binary (4)</td>
</tr>
<tr>
<td>2</td>
<td>Operation identifier</td>
<td>Input</td>
<td>Binary (4)</td>
</tr>
<tr>
<td>3</td>
<td>User profile</td>
<td>Input</td>
<td>Char (10)</td>
</tr>
<tr>
<td>4</td>
<td>Remote IP address</td>
<td>Input</td>
<td>Char (10)</td>
</tr>
<tr>
<td>5</td>
<td>Length of remote IP address</td>
<td>Input</td>
<td>Binary (4)</td>
</tr>
<tr>
<td>6</td>
<td>Operation-specific information</td>
<td>Input</td>
<td>Char (*)</td>
</tr>
<tr>
<td>7</td>
<td>Length of operation-specific information</td>
<td>Input</td>
<td>Binary (4)</td>
</tr>
<tr>
<td>8</td>
<td>Allow operation</td>
<td>Output</td>
<td>Binary (4)</td>
</tr>
</tbody>
</table>

Here are the parameter descriptions

VLRQ0100 Parameter 1: Application identifier

**INPUT; BINARY(4)**

Identifies the TCP/IP application program that is making the request. Four different TCP/IP applications share the VLRQ0100 interface. The first parameter identifies which application is calling the exit program.
The possible values are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FTP client program</td>
</tr>
<tr>
<td>1</td>
<td>FTP server program</td>
</tr>
<tr>
<td>2</td>
<td>REXEC server program</td>
</tr>
<tr>
<td>3</td>
<td>TFTP server program</td>
</tr>
</tbody>
</table>

**VLRQ0100 Parameter 2:**

**Operation identifier**

**Input:** Binary(4)

Indicates the operation (command) that the FTP user wants (requests) to perform.

When the application identifier (parameter 1) indicates the FTP client or FTP server program, the possible values are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Start session</td>
</tr>
<tr>
<td>1</td>
<td>Create directory/library*</td>
</tr>
<tr>
<td>2</td>
<td>Delete directory/library*</td>
</tr>
<tr>
<td>3</td>
<td>Set current directory/library</td>
</tr>
<tr>
<td>4</td>
<td>List files *</td>
</tr>
<tr>
<td>5</td>
<td>Delete file *</td>
</tr>
<tr>
<td>6</td>
<td>Send file</td>
</tr>
<tr>
<td>7</td>
<td>Receive file</td>
</tr>
<tr>
<td>8</td>
<td>Rename file*</td>
</tr>
<tr>
<td>9</td>
<td>Execute CL command</td>
</tr>
</tbody>
</table>

**Note:** The symbol * represents control operations that the FTP client exit does not recognize. The only way a client can use these operations is with CL commands using the FTP client subcommand SYSCMD. Operation identifier 9 controls the execution of CL commands.

**VLRQ0100 Parameter 3:**

**User profile**

**INPUT:** Char(10)

The user profile for the FTP session.

**VLRQ0100 Parameter 4:**

**Remote IP address**

**INPUT:** CHAR(*)

The Internet Protocol (IP) address of the remote host system. The format for this string is dotted decimal (123.45.67.89), left justified. The remote host may be a client or a server that is based on the setting of the application identifier parameter.

**VLRQ0100 Parameter 5:**

**Length (in bytes) of the remote IP address (parameter 4)**

**INPUT:** BINARY(4)

The length of the remote IP address (parameter 4).

**VLRQ0100 Parameter 6:**

**Operation-specific information**
INPUT; CHAR(*)
Information that describes the requested operation. The contents of this field depend on the values of the operation identifier (parameter 2), and the application identifier (parameter 1). For example:

For operation identifier 0 and application identifier 0
There is no operation-specific information. This field is blank.

For operation identifier 0 and application identifier 1
The operation-specific information contains the IP address of the TCP/IP interface that connects to the local host (FTP server) for this session. The format for this string is dotted decimal (123.45.67.89), left justified.

For operation identifiers 1 through 3
The operation-specific information contains the name of the directory or library in which to perform the operation. The format for the directory or library name is an absolute path name.

For operation identifiers 4 through 8
The operation-specific information contains the name of the file on which to perform the operation. The format for the file name is an absolute path name.

For operation identifier 9
The operation-specific information contains the iSeries 400 Control Language (CL) command the user requests.

Note: See also: VLRQ0100 exit point format usage notes.

VLRQ0100 Parameter 7:
Length of operation-specific information

INPUT; BINARY(4)
Indicates the length of the operation-specific information (parameter 6). Length is 0 when the exit point does not provide operation-specific information.

VLRQ0100 Parameter 8:
Allow operation

OUTPUT; BINARY(4)
Indicates whether to allow or reject the requested operation.

The possible values are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Never allow this operation identifier: Reject this operation identifier unconditionally for the remainder of the current session. This operation identifier will not call the exit program again.</td>
</tr>
<tr>
<td>0</td>
<td>Reject the operation</td>
</tr>
<tr>
<td>1</td>
<td>Allow the operation</td>
</tr>
<tr>
<td>2</td>
<td>Always allow this operation identifier: Allow this operation identifier unconditionally for the remainder of the current session. This operation identifier will not call the exit program again.</td>
</tr>
</tbody>
</table>
VRQ0100 exit point format usage notes: VRQ0100 is the exit point format that is used for both the FTP Client Request Validation Exit Point and the FTP Server Request Validation Exit Point.

Invalid Output Parameters

If the output returned for the Allow Operation parameter (parameter 8) is not valid, then the FTP server rejects the requested operation and posts this message to the job log:

Data from exit program for exit point &1 is missing or not valid

Exceptions

If the FTP server encounters any exception when calling the exit program, it posts this message to the job log:

Exception encountered for FTP exit program &1 in library &2 for exit point &3

Summary: Operation-specific information

This table summarizes the Operation-specific information (VRQ0100 parameter 6) that is required for each Operation identifier (VRQ0100 parameter 2).

<table>
<thead>
<tr>
<th>Operation Identifier (VRQ0100 Parm 2)</th>
<th>Operation-Specific Information (VRQ0100 Parameter 6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NONE if application ID=0 (parameter 1)</td>
</tr>
<tr>
<td></td>
<td>Dotted decimal format IP address of client host when application ID=1 or 2 (parameter 1)</td>
</tr>
<tr>
<td>1-3</td>
<td>Absolute path name of library or directory. Examples: /QSYS.LIB/QGPL.LIB (a) /QOpenSys/DirA/DirAB/DirABC (b)</td>
</tr>
<tr>
<td>4-8</td>
<td>Absolute path name of file. Examples: /QSYS.LIB/MYLIB.LIB/FILE.MEMB.MBR (a) /QOpenSys/DirA/DirAB/FileA1 (b)</td>
</tr>
</tbody>
</table>

Notes:

(a) QSYS.LIB file system path names are always in uppercase
(b) QOpenSys file system path names are case sensitive and may include upper and lower case letters.

Logon exit point for TCP/IP application servers: You can control the authentication of users to a TCP/IP application server with the TCP/IP Application Server Logon Exit Point. This exit point allows server access based on the originating session’s address. It also allows you to specify an initial working directory that is different from those that are in the user profile.

When you add an exit program to the exit point, the server calls the logon exit program each time a user attempts to log on. The exit program sets the return code output parameter to indicate whether or not the server will continue the logon operation. Alternate return code settings are available for processing the logon, and initializing directory information.

The iSeries 400 exit point for FTP server logon is:

QIBM_QTMF_SVR_LOGON

These are the three exit point formats available:

- The TCPL0100 exit point format allows this basic logon control:
  - Ability to accept or reject a logon
  - Control of the user profile, password, and current library
• The TCPL0200 exit point format provides additional parameters to control the logon process, including:
  – Ability to set the working directory to any directory on the system.
  – Ability to return application-specific information
  – Ability to control encryption of FTP data sent to and received from the FTP client.
• The TCPL0300 exit point format extends the TCPL0200 format to allow usage of OS/400 enhanced password support and provides additional parameters to allow CCSID processing for password and directory name fields. In addition, when the user for the session has been authenticated with a client certificate, the client certificate is provided to the exit program.

Notes:
1. There can be only one exit program registered for the FTP server logon exit point. You must decide which of the three exit point formats you want to use.
2. For the FTP application, this exit point provides the capability to implement anonymous FTP, including the information required to log and control access.
3. For all character parameters in exit point formats TCPL0100 and TCPL0200, and all character parameters without an associated CCSID in exit point format TCPL0200: Character data passed to the exit program is in the CCSID of the job. If the job CCSID is 65535, the character data is in the default CCSID of the job. Any character data that is returned by the exit program in these parameters is expected to be in this same CCSID.

QTCP Needs Authority

When the application calls the FTP server logon exit program, the FTP server job is running under the QTCP user profile.

Make sure that QTCP has sufficient authority to access and write to any log files or other satellite files associated with the exit programs.

For examples of the server logon exit point program, see the following:

Example: FTP server logon exit program in C code
Example: FTP server logon exit program in CL code
Example: FTP server logon exit program in ILE RPG code

Related information:
• Logon exit point for TCP/IP application servers
• FTP Server Logon exit point program

TCPL0100 exit point format:
Exit Point Format Name: TCPL0100
Exit Point Name: QIBM_QTMF_SVR_LOGON
Exit Point Name: QIBM_QTMX_SVR_LOGON

This is the required parameter group:

<table>
<thead>
<tr>
<th></th>
<th>Application identifier</th>
<th>Input</th>
<th>Binary(4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>User identifier</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>3</td>
<td>Length of user identifier</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>4</td>
<td>Authentication string</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td></td>
<td>Length of authentication string</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>---</td>
<td>---------------------------------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>6</td>
<td>Client IP address</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>7</td>
<td>Length of client IP address</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>8</td>
<td>Return code</td>
<td>Output</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>9</td>
<td>User profile</td>
<td>Output</td>
<td>Char(10)</td>
</tr>
<tr>
<td>10</td>
<td>Password</td>
<td>Output</td>
<td>Char(10)</td>
</tr>
<tr>
<td>11</td>
<td>Initial current library</td>
<td>Output</td>
<td>Char(10)</td>
</tr>
</tbody>
</table>

**Parameter Descriptions**

**Application identifier**

INPUT; BINARY(4) Identifies the requested application server. The valid values are:

1 FTP server program
2 REXEC server program

**User identifier**

INPUT; CHAR(*) The user identification supplied by the client program.

For the FTP server, this parameter contains the data field from the USER subcommand.

**Length of user identifier**

INPUT; BINARY(4) The length (in bytes) of the user identifier string.

**Authentication string**

INPUT; CHAR(*) The string (such as a password) supplied by the client program.

For the FTP server, this parameter contains the data field from the PASS (password) subcommand. Beginning with V5R1, if the user is authenticated via a client certificate, no data is provided for this parameter.

**Length of authentication string**

INPUT; BINARY(4) The length (in bytes) of the authentication string.

**Note:** For the FTP server: When the user is authenticated via a client certificate, this parameter is set to 0.

**Client IP address**

INPUT; CHAR(*) The Internet Protocol (IP) address from which the session originates. This string is in dotted decimal format, left justified.

**Length of client IP address**

INPUT; BINARY(4) Indicates the length (in bytes) of the client IP address.

**Return code**

OUTPUT; BINARY(4) Indicates whether to accept or reject the logon operation, to perform password authentication, and whether or not to override the initial current library. The valid values are:

0 Reject the logon operation. Ignore the user profile, password, and initial current library output parameters.
1 Continue the logon operation with the specified user identifier and authentication string, and the user-specified initial current library. The user identifier becomes the user profile, and the authentication string becomes the password. The program ignores the user profile.
profile, password, and initial current library output parameters.

Note: For the logon to succeed, the authentication string must match the user profile-specified password.

2 Continue the logon operation with the specified user identifier and authentication string, and override the initial current library with the one specified by the initial current library parameter. The user identifier is the user profile. The authentication string is the password. Provide the initial current library output parameter. The program ignores the user profile and password output parameters.

Note: For the logon to succeed, the authentication string must match the user profile-specified password.

3 Continue the logon operation. Override the user profile and password with those values you received from the output parameters of this exit program. Use the user profile-specified initial current library that the exit program returns. The program ignores the initial current library output parameter.

Note: For the logon to succeed, the password output parameter must match the user profile-specified password. Attention! IBM strongly recommends that you never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

4 Continue the logon operation, which will override the user profile, password, and initial current library with output parameters of this exit program.

Note: For the logon to succeed, the password output parameter must match the user profile-specified password. Attention! IBM strongly recommends that you never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

5 Accept the logon operation. Override the user profile is returned in the user profile output parameter of this exit program. Use the initial current library specified by the user profile, returned by this exit program. The program ignores the output parameters for the initial current library and password.

Note: Specifying this value will override normal OS/400 password processing. It is the only password authentication.

6 Accept the logon operation. Override the user profile and initial current library with those that are returned in the output parameters of this exit program. Ignore the output parameter for password.

Note: Specifying this value will override normal OS/400 password processing. It is the only password authentication.

User profile
OUTPUT; CHAR(10) The user profile to use for this session. This parameter must be left justified and padded with blanks.
Password
OUTPUT; CHAR(10) The password to use for this session. This parameter must be left justified and padded with blanks.

Initial current library
OUTPUT; CHAR(10) The initial current library to be established for this session. This parameter must be left justified and padded with blanks.

Go To:
• TCPL0100 Format Usage Notes [TCPL0100 Format Usage Notes]

You may also need to refer to:
• TCPL0200 Format [TCPL0200 exit point format] on page 17: This exit point provides additional parameters to control the logon process.
• TCPL0300 Format: This exit point identifies the application server from which the request is being made.
• TCP/IP Application Server Logon Exit Point Interface [Logon exit point for TCP/IP application servers] on page 12: You can control the authentication of users to a TCP/IP application server with this TCP/IP Application Server Logon Exit Point.

TCPL0100 Format Usage Notes: For FTP, if any of the returned output parameters are not valid, the FTP server will not allow the operation. In this case, the FTP server issues the message Data from exit program for exit point &1 is missing or not valid to the job log.

For FTP, if you encounter any exception when you call the exit program, the FTP server issues this message: Exception encountered for FTP exit program &1 in library &2 for exit point &3

This table summarizes what the FTP server will do, depending on the value of the return code (parameter 8) that is returned to the FTP server by the exit program.

<table>
<thead>
<tr>
<th>Return Code</th>
<th>User Profile (9)</th>
<th>Password (10)</th>
<th>Initial Lib (11)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td>1</td>
<td>(User identifier, parameter 2)</td>
<td>(Password, parameter 4)</td>
<td>(From user profile)</td>
</tr>
<tr>
<td>2</td>
<td>(User identifier, parameter 2)</td>
<td>(Password, parameter 4)</td>
<td>Return value</td>
</tr>
<tr>
<td>3</td>
<td>Return value</td>
<td>Return value</td>
<td>Return value</td>
</tr>
<tr>
<td>4</td>
<td>Return value</td>
<td>Return value</td>
<td>Return value</td>
</tr>
<tr>
<td>5</td>
<td>Return value</td>
<td>Ignored</td>
<td>(From user profile)</td>
</tr>
<tr>
<td>6</td>
<td>Return value</td>
<td>Ignored</td>
<td>Return value</td>
</tr>
</tbody>
</table>

In the table above, the values in parentheses indicate what the TCP/IP application uses for information when it ignores the output value. The entry Ignored means that it used no value; therefore return nothing for that return code value.

For the FTP server (exit point QIBM_QTMF_SVR_LOGON, application identifier 1): when the user identifier is ANONYMOUS and this exit point adds the exit program, the server issues this special reply when requesting the password: 331 Guest logon in process, send complete e-mail address as password. The application issues this message before calling the exit program.

After the application accepts the server logon, the FTP server issues this reply: 230 Guest logon accepted, access restrictions apply
For the REXEC server (application identifier 2):
1. If the return allow operation output parameter is not valid, the REXEC server will not allow the operation. The REXEC server issues the message "Data from exit program for exit point &1 is missing or not valid" to the job log.
2. If the REXEC server encounters any exception when calling the exit program, the REXEC server will not allow the operation. It issues the message "Exception encountered for REXEC exit program &1 in library &2 for exit point &3," to the job log.

TCPL0200 exit point format:
Exit Point Format Name: TCPL0200
Exit Point Name: QIBM_QTMF_SVR_LOGON

This is the required parameter group:

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Input/Output</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application identifier</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>2</td>
<td>User identifier</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>3</td>
<td>Length of user identifier</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>4</td>
<td>Authentication string</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>5</td>
<td>Length of authentication string</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>6</td>
<td>Client IP address</td>
<td>Input</td>
<td>Char(*)</td>
</tr>
<tr>
<td>7</td>
<td>Length of client IP address</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>8</td>
<td>Allow logon</td>
<td>Output</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>9</td>
<td>User profile</td>
<td>Output</td>
<td>Char(10)</td>
</tr>
<tr>
<td>10</td>
<td>Password</td>
<td>Output</td>
<td>Char(10)</td>
</tr>
<tr>
<td>11</td>
<td>Initial current library</td>
<td>Input/Output</td>
<td>Char(10)</td>
</tr>
<tr>
<td>12</td>
<td>Initial home directory</td>
<td>Output</td>
<td>Char(*)</td>
</tr>
<tr>
<td>13</td>
<td>Length of initial home directory</td>
<td>Input/Output</td>
<td>Binary(4)</td>
</tr>
<tr>
<td>14</td>
<td>Application-specific information</td>
<td>Input/Output</td>
<td>Char(*)</td>
</tr>
<tr>
<td>15</td>
<td>Length of application-specific information</td>
<td>Input</td>
<td>Binary(4)</td>
</tr>
</tbody>
</table>

Parameter Descriptions

Application identifier
INPUT; BINARY(4) Identifies the application server from which the request is being made. The valid values are:
1. FTP server program

User identifier
INPUT; CHAR(*) The user identification supplied by the client program.
For the FTP server, this parameter contains the data field from the USER subcommand.

Length of user identifier
INPUT; BINARY(4) The length (in bytes) of the user identifier string.

Authentication string
INPUT; CHAR(*) The string (such as a password) supplied by the client program.
For the FTP server, this parameter contains the data field from the PASS (password) subcommand. Beginning with V5R1, if the user is authenticated via a client certificate, no data is provided for this parameter.
Length of authentication string
INPUT; BINARY(4) The length (in bytes) of the authentication string.

Note: For the FTP server: When the user is authenticated via a client certificate, this parameter is set to 0.

Client IP address
INPUT; CHAR(*) The Internet Protocol (IP) address from which the session originates. This string is in dotted decimal format, left justified.

Length of client IP address
INPUT; BINARY(4) Indicates the length (in bytes) of the client IP address.

Allow logon
OUTPUT; BINARY(4) Indicates whether the logon operation should be accepted or rejected, and how password authentication is performed. The valid values are:

0  Reject the logon operation. Ignores all other output parameters.
1  Continue the logon operation with the specified user identifier and authentication string. The user identifier is the user profile, and the authentication string is the password. The current library and working directory is based on the settings of those output parameters. The application ignores the user profile and password output parameters.

Note: For the logon to succeed, the authentication string must match the user profile-specified password.

2  Continue the logon operation. Override the user profile and password with the returned values in the output parameters of this exit program. The application initializes the current library and working directory based on the settings of those output parameters.

Note: For the logon to succeed, the password output parameter must match the user profile-specified password.

Attention! IBM strongly recommends that you never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

3  Accept the logon operation. Override the user profile with the profile returned in the user profile output parameter of this exit program. The program initializes the current library and working directory based on the settings of the output parameters. It ignores the password output parameter.

Note: If your system is running at a security level of 20 or higher, specifying this value overrides normal OS/400 password processing. This is the only password authentication.

User profile
OUTPUT; CHAR(10) The user profile to use for this session. When required, this parameter must be left justified and padded with blanks.

Password
OUTPUT; CHAR(10) The password to use for this session. When required, this parameter must be left justified and padded with blanks.
Initial current library
OUTPUT; CHAR(10) The initial current library to use for this session. When required, this parameter must be left justified and padded with blanks. This parameter is set to the following special value when the exit program is called:

*CURLIB
Use the current library that the user profile specifies.

Initial home directory
OUTPUT; CHAR(*) The initial setting of the home directory to use for this session. When specified, this parameter must be a valid absolute path name, and the length of initial home directory parameter set to the proper value.

Length of initial home directory
INPUT/OUTPUT; BINARY(4) The length of the initial home directory parameter returned by the exit program. This parameter initializes at zero when the application calls the exit program. If the exit program does not change the value of the parameter, the home directory is initialized to the home directory that the user’s profile specifies.

Application-specific information
INPUT/OUTPUT; CHAR(*) Information that is used to communicate application-specific logon settings. For the correct format, see Format of application-specific information parameter.

Length of application-specific information
INPUT; BINARY(4) The length (in bytes) of the application-specific information.

You may also need to refer to:
• TCP/IP Application Server Logon Exit Point Interface: You can control the authentication of users to a TCP/IP application server with this TCP/IP Application Server Logon Exit Point.

TCP0300 exit point format:
Exit Point Format Name: TCPL0300
Exit Point Name: QIBM_QTMF_SVR_LOGON
Exit Point Name: QIBM_QTMX_SVR_LOGON

This is the required parameter group:

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application identifier</td>
<td>Input</td>
</tr>
<tr>
<td>2</td>
<td>User identifier</td>
<td>Input</td>
</tr>
<tr>
<td>3</td>
<td>Length of user identifier</td>
<td>Input</td>
</tr>
<tr>
<td>4</td>
<td>Authentication string</td>
<td>Input</td>
</tr>
<tr>
<td>5</td>
<td>Length of authentication string</td>
<td>Input</td>
</tr>
<tr>
<td>6</td>
<td>CCSID of authentication string</td>
<td>Input</td>
</tr>
<tr>
<td>7</td>
<td>Client IP address</td>
<td>Input</td>
</tr>
<tr>
<td>8</td>
<td>Length of client IP address</td>
<td>Input</td>
</tr>
<tr>
<td>9</td>
<td>Allow logon</td>
<td>Output</td>
</tr>
<tr>
<td>10</td>
<td>User profile</td>
<td>Output</td>
</tr>
<tr>
<td>11</td>
<td>Password</td>
<td>Output</td>
</tr>
<tr>
<td>12</td>
<td>Length of password</td>
<td>Output</td>
</tr>
<tr>
<td>13</td>
<td>CCSID of password</td>
<td>Output</td>
</tr>
<tr>
<td>14</td>
<td>Initial current library</td>
<td>Input/Output</td>
</tr>
<tr>
<td></td>
<td>Parameter Descriptions</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td><strong>Initial home directory</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td><strong>Length of initial home directory</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input/Output</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td><strong>CCSID of initial home directory</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input/Output</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td><strong>Application-specific information</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input/Output</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td><strong>Length of application-specific information</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td></td>
</tr>
</tbody>
</table>

**Parameter Descriptions**

**Application identifier**

*INPUT; BINARY(4)* Identifies the application server from which the request is being made. The valid values are:

1 FTP server program

2 REXEC server program

**User identifier**

*INPUT; CHAR(*)* The user identification supplied by the client program.

For the FTP server, this parameter contains the data field from the USER subcommand.

**Length of user identifier**

*INPUT; BINARY(4)* The length (in bytes) of the user identifier string.

**Authentication string**

*INPUT; CHAR(*)* The string (such as a password) supplied by the client program.

For the FTP server, this parameter contains the data field from the PASS (password) subcommand (unless the user is authenticated via a client certificate, in which case the client certificate is provided for this parameter).

**Length of authentication string**

*INPUT; BINARY(4)* The length (in bytes) of the authentication string.

**CCSID of authentication string**

*INPUT; BINARY(4)* The CCSID of the authentication string parameter.

For the FTP server: When the user is authenticated via a client certificate, this parameter is set to -2.

**Client IP address**

*INPUT; CHAR(*)* The Internet Protocol (IP) address from which the session originates. This string is in dotted decimal format, left justified.

**Length of client IP address**

*INPUT; BINARY(4)* Indicates the length (in bytes) of the client IP address.

**Allow logon**

*OUTPUT; BINARY(4)* Indicates whether the logon operation should be accepted or rejected, and how password authentication is performed. The valid values are:

0 Reject the logon operation. Ignores all other output parameters.

1 Continue the logon operation with the specified user identifier and authentication string. The user identifier is the user profile, and the authentication string is the password. The current library and working directory is based on the settings of those output parameters. The application ignores the user profile and password output parameters.
Note: For the logon to succeed, the authentication string must match the user profile-specified password.

2 Continue the logon operation. Override the user profile and password with the returned values in the output parameters of this exit program. The application initializes the current library and working directory based on the settings of those output parameters.

Note: For the logon to succeed, the password output parameter must match the user profile-specified password. **Attention!** IBM strongly recommends that you **never** code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

3 Accept the logon operation. Override the user profile with the profile returned in the user profile output parameter of this exit program. The program initializes the current library and working directory based on the settings of the output parameters. It ignores the password output parameter.

Note: If your system is running at a security level of 20 or higher, specifying this value overrides normal OS/400 password processing. This is the only password authentication.

**User profile**

OUTPUT; CHAR(10) The user profile to use for this session. When required, this parameter must be left justified and padded with blanks.

**Password**

OUTPUT; CHAR(*) The password to use for this session. When required, the length of password and CCSID of password parameters must also be specified, and this parameter must be left-justified. When the QPWLVLVL system value is set to 0 or 1, up to 10 characters may be specified; when the QPWLVLVL system value is set to 2 or 3, up to 128 characters may be specified.

**Length of password**

OUTPUT; BINARY(4) The length (in bytes) of the password. When required, the valid range is 1 to 512 bytes.

**CCSID of password**

OUTPUT; BINARY(4) The CCSID of the password. This parameter must be set by the exit program when the password parameter is specified. The valid values are:

0 The CCSID of the job is used to determine the CCSID of the data to be converted. If the job CCSID is 65535, the CCSID from the default CCSID (DFTCCSID) job attribute is used.

1-65533 A valid CCSID in this range.

**Initial current library**

OUTPUT; CHAR(10) The initial current library to use for this session. When required, this parameter must be left justified and padded with blanks. This parameter is set to the following special value when the exit program is called:

*CURLIB

Use the current library that the user profile specifies.

**Initial home directory**

OUTPUT; CHAR(*) The initial setting of the home directory to use for this session. When specified,
this parameter must be a valid absolute path name, and the length of initial home directory and CCSID of initial home directory parameters set to the proper values.

**Length of initial home directory**

INPUT/OUTPUT; BINARY(4) The length of the initial home directory parameter returned by the exit program. This parameter initializes at zero when the application calls the exit program. If the exit program does not change the value of the parameter, the home directory is initialized to the home directory that the user’s profile specifies.

**CCSID of initial home directory**

OUTPUT; BINARY(4) The CCSID of the initial home directory. This parameter must be set by the exit program when the initial home directory is specified. The valid values are:

0 The CCSID of the job is used to determine the CCSID of the data to be converted. If the job CCSID is 65535, the CCSID from the default CCSID (DFTCCSID) job attribute is used.

1-65533 A valid CCSID in this range.

Application-specific information

INPUT/OUTPUT; CHAR(*) Information that is used to communicate application-specific logon settings. For the correct format, see Format of application-specific information parameter.

**Length of application-specific information**

INPUT; BINARY(4) The length (in bytes) of the application-specific information.

You may also need to refer to:

- TCP/IP Application Server Logon Exit Point Interface [Logon exit point for TCP/IP application servers](#) on page 12: You can control the authentication of users to a TCP/IP application server with this TCP/IP Application Server Logon Exit Point.

Example: FTP Server Logon exit program in C code: This is an example of a simple FTP Server Logon exit program in C programming language.

Note: These examples are for illustration purposes only. They do not contain enough features to run on a production machine as is. Feel free to use them as a starting point, or to copy sections of code them from as you write your own programs. Once you write your programs, you must test the code.

Preformatted text in the following example will flow outside the frame.
/* Module Description ***********************************************************/
/* */
/* DISCLAIMER: This material contains programming source code */
/* for your consideration. These examples have not been thoroughly */
/* tested under all conditions. IBM, therefore, cannot guarantee or */
/* imply reliability, serviceability, performance or function of */
/* these programs. All programs contained herein are provided to */
/* you "AS IS". THE IMPLIED WARRANTIES OF MERCHANTABILITY AND */
/* FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY DISCLAIMED. */
/* */
/* LIMITATION OF LIABILITY: Neither IBM nor the author shall be */
/* liable for any claims or damages whatsoever, including property */
/* damage, personal injury, intellectual property infringement, loss */
/* of profits, or interruption of business, or for any special, */
/* consequential or incidental damages, however caused, whether */
/* arising out of breach of warranty, contract, tort (including */
/* negligence), strict liability, or otherwise. */
/* */
/******************************************************************************/
/**/ 
/* Source File Name: qtmfsvrln.c */
/* */
/* Module Name: FTP Server Logon exit program. */
/* */
/* Service Program Name: n/a */
/* */
/* Source File Description: */
/* This example exit program provides additional control over the */
/* process of authenticating a user to a TCP/IP application server. */
/* When installed, this example exit program would be called each */
/* time a user attempts to log on to the server. */
/* */
/******************************************************************************/
/* Function List: main - FTP Server Logon exit program main. */
/* qtmfsvrln - FTP Server Logon exit function. */
/* CheckClientAddress - Check originating sessions IP */
/* address. */
/* */
/* End Module Description ******************************************************/
#define _QTMFSVRLGN_C

/******************************************************************************/
/* All file scoped includes go here */
/******************************************************************************/
ifndef __stdio_h
#include <stdio.h>
endif

define __ctype_h
#include <ctype.h>
endif

define __string_h
#include <string.h>
endif

define __stdlib_h
#include <stdlib.h>
endif

#include "qusec.h" /* Include for API error code structure */
#include "qsyrusri.h" /* Include for User Information API */

/******************************************************************************/
/* All file scoped Constants go here */
/******************************************************************************/
define EQ  ==

Example: FTP Server Logon exit program in CL code:  This is an example of a simple FTP Server Logon exit program in the iSeries 400 Command Language (CL). This is for exit point format TCPL0100.

Note:  These examples are for illustration purposes only. They do not contain enough features to run on a production machine as is. Feel free to use them as a starting point, or to copy sections of code from them as you write your own programs. Once you write the program, you must test them.
Sample FTP server logon exit program.

Note: This program is a sample only and has not undergone any formal review or testing.

Additional notes:
1. When the FTP server logon exit is called, the FTP server job is running under the QTCP user profile.
2. For the ANONYMOUS case, users can add logging capability (for example, write the E-mail address entered for the password and the client IP address to a log file).
3. IBM strongly recommends that you create the exit program in a library with *PUBLIC authority set to *EXCLUDE, and give the exit program itself a *PUBLIC authority of *EXCLUDE. The FTP server adopts authority when it is necessary to resolve and call the exit program.

TSTLOGCL: PGM PARM(&APPIDIN &USRIN &USRLENIN &AUTIN &AUTLENIN &IPADDRIN &IPLENIN &RETCDOUT &USRPRFOUT &PASSWDOUT &CURLIBOUT);

/* Declare input parameters */
DCL VAR(&APPIDIN); TYPE(*CHAR) LEN(4) /* Application identifier */
DCL VAR(&USRIN); TYPE(*CHAR) LEN(999) /* UserID */
DCL VAR(&USRLENIN); TYPE(*CHAR) LEN(4) /* Length of userID */
DCL VAR(&AUTIN); TYPE(*CHAR) LEN(999) /* Authentication string */
DCL VAR(&AUTLENIN); TYPE(*CHAR) LEN(4) /* Length of auth. string */
DCL VAR(&IPADDRIN); TYPE(*CHAR) LEN(15) /* Client IP address */
DCL VAR(&IPLENIN); TYPE(*CHAR) LEN(4) /* IP address length */
DCL VAR(&RETCDOUT); TYPE(*CHAR) LEN(4) /* return code (out) */
DCL VAR(&USRPRFOUT); TYPE(*CHAR) LEN(10) /* user profile (out) */
DCL VAR(&PASSWDOUT); TYPE(*CHAR) LEN(10) /* password (out) */
DCL VAR(&CURLIBOUT); TYPE(*CHAR) LEN(10) /* current library (out) */

/* Declare local copies of parameters (in format usable by CL) */
DCL VAR(&APPID); TYPE(*DEC) LEN(10)
DCL VAR(&USRLEN); TYPE(*DEC) LEN(5 0)
DCL VAR(&AUTLEN); TYPE(*DEC) LEN(5 0)
DCL VAR(&IPLEN); TYPE(*DEC) LEN(5 0)

/* Assign input parameters to local copies */
CHGVAR VAR(&APPID); VALUE(%BINARY(&APPIDIN));
CHGVAR VAR(&USRLEN); VALUE(%BINARY(&USRLENIN));
CHGVAR VAR(&AUTLEN); VALUE(%BINARY(&AUTLENIN));
CHGVAR VAR(&IPLEN); VALUE(%BINARY(&IPLENIN));

/* Check for ANONYMOUS user. Allow for ANONYMOUS, and so forth as "regular" user profile */
IF COND(&USRLEN = 9) THEN(DO)
  IF COND(%SST(&USRIN 1 9) = 'ANONYMOUS') THEN(DO)
    /* For anonymous user: want to force user profile ANONYMOUS current library to PUBLIC. */
    /* */
Example: FTP Server Logon exit program in ILE RPG code: This is an example of a simple FTP Server Logon exit program in ILE RPG.

Note: These examples are for illustration purposes only. They do not contain enough features to run on a production machine as is. Feel free to use them as a starting point, or to copy sections of code from them as you write your own programs. Once you write the programs, you must test them.

Preformatted text in the following example will flow outside the frame.
Module Description

DISCLAIMER: This material contains programming source code for your consideration. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, performance or function of these programs. All programs contained herein are provided "AS IS". THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY DISCLAIMED.

LIMITATION OF LIABILITY: Neither IBM nor the author shall be liable for any claims or damages whatsoever, including property damage, personal injury, intellectual property infringement, loss of profits, or interruption of business, or for any special, consequential or incidental damages, however caused, whether arising out of breach of warranty, contract, tort (including negligence), strict liability, or otherwise.

PROGRAM FUNCTION

This program demonstrates some of the abilities an FTP Server Logon Exit Program can have.

INDICATOR USAGE

IND. DESCRIPTION

LR - CLOSE FILES ON EXIT

DATA STRUCTURES USED BY THIS PROGRAM

Define constants

D Anonym C CONST('ANONYMOUS ')
D Text1 C CONST('Anonymous (')
D Text2 C CONST(') FTP logon')
D InvalidNet C CONST('10.')

VARIABLE DEFINITIONS AND LISTS USED BY THIS PROGRAM

Define binary parameters

D APPIDds S 10I 0
D USRLENds S 10I 0
D AUTLENds S 10I 0
D IPLENds S 10I 0
D RETCDds S 10I 0

Define parameter list

C *Entry PLIST

Input parameters:

C * PARM APPIDIN Application ID possible values: 1 = FTP Server Program
C * PARM USRIN User ID
C * PARM USRLENIN Length of User ID
Removing exit programs: To remove an installed exit program:

1. Enter **WRKREGINF** at an iSeries 400 command line.
2. Page down to an FTP Server Logon exit point:
   
   QIBM_QTMF_SERVER_REQ VRLOO100  
   QIBM_QTMF_SRV_LOGON TCPL0100  
   QIBM_QTMF_SRV_LOGON TCPL0200  
   QIBM_QTMF_SRV_LOGON TCPL0300  

3. Enter 8 in the Opt field to the left of the exit point entry and press **Enter**.
4. At the Work with Exit Program display, enter a 4 (Remove).
5. Enter the name of the exit program in Exit Program field.
6. Enter the name of the library that contains the exit program in the Library field.
7. Press **Enter**.
8. After you finish removing exit points, stop and restart the FTP server.

Improving FTP server performance with configurable subsystem support

You can specify a separate subsystem for the FTP server. This should increase performance, because the need to share resources is eliminated.

Use the CHGFTPA command to configure a subsystem for the FTP server. If the specified subsystem does not exist, then FTP will create it along with routing table entries and job descriptions. When the startup job for the server is executed, it will specify the parameters for the newly created subsystem and then submit the server jobs for batch startup in that subsystem.

Working with the FTP client

The FTP client allows you to transfer files that are found on your iSeries 400 system, including those in the Root, QSYS.Lib, QOpenSys, QOPT, and QFileSvr.400 file systems. It also allows you to transfer folders and documents in the document library services (QDLS) file system. The FTP client may be run interactively or unattended in batch. It also includes other features for manipulating files on your system.

The client has a user interface from which you can enter client subcommands for making requests to an FTP server. The results of these requests are then displayed.

One may interact with the client online or run the FTP client in an unattended batch mode where client subcommands are read from a file and the responses to these subcommands are written to a file.

To transfer files between the client and the server, two connections are established. The control connection is used to request services from the server with FTP server commands. The server sends replies back to the client to indicate how the request was handled. The second connection, called the data connection, is used for transferring lists of files and the actual file data.

Both the client and the server have a data transfer function that interfaces to the resident file systems. These functions read or write data to the local file systems and to and from the data connection.

- Starting and ending a client session
  Learn how to start and stop a client session.
- File systems and naming conventions
  Become familiar with the naming conventions that the iSeries server uses.
- Transferring files with FTP
  Learn how to send and receive files with FTP.
• FTP as batch job
  View examples of how to run FTP in an unattended mode.
• FTP client subcommands
  See syntax and description information on the FTP client subcommands that are used to instruct the
  FTP client to establish connections, navigate libraries and directories, create files, delete files, and
  transfer files.

If you are having problems, see Troubleshooting FTP for information on troubleshooting the FTP server and
client.

Starting and ending a client session

Starting the FTP client session
The CL command FTP starts a client session on the local iSeries 400 system and then opens a
connection to the FTP server on the specified remote system. For example, entering the CL command
“FTP myserver.com” would start a client session on your iSeries 400 system, then open a connection to
the FTP server on the remote “myserver.com” system.

Ending the FTP client session
The FTP session is ended with the QUIT subcommand. The QUIT subcommand closes the connection
with the remote host and ends the FTP session on the iSeries 400 server. Alternatively, you can press F3
(Exit) and then confirm

File systems and naming conventions
The FTP server arranges the information units of a file system in a multiple-level tree-like structure.

The OS/400 file systems that are supported by FTP vary depending on the release level of your iSeries
400 server. File systems on OS/400 can use different terms for data and the hierarchical grouping of data.

Naming conventions

Each OS/400 file system has its own set of rules for naming files. The format used to name any file must
adhere to the naming conventions of the file system in which it resides. Formats and examples of file
names for FTP-supported OS/400 file systems are described in IFS. Refer to the File Systems and
Management article for more information. The server may provide naming information for files on
non-iSeries 400 servers when you use QUOTE HELP "QUOTE (Send a Subcommand to an FTP Server)"
on page 57.

See the NAMEFMT (Select File Naming Format) "NAMEFMT (Select File Naming Format)" on page 54
client subcommand page for instructions on the use of the NAMEFMT subcommand to work with file name
formats.

Transferring files with FTP
Follow these steps to transfer files with FTP.
1. Collect this information:
   • The TCP/IP name or IP address of the remote computer
   • A logon name and password for the remote computer (unless the remote computer supports
     anonymous FTP)
   • The name and location of the file you want to transfer
   • The location of the destination
   • The file transfer type "File transfer types" on page 30 that you will use: ASCII, EBCDIC, or BINARY
2. Enter FTP on the command line. Press the Enter key.
3. At the prompt, enter the TCP/IP name or IP address of the remote computer system. You can use either the name or the IP address, such as:

remote.systemname.com

or

110.25.9.13

4. Enter the Coded Character Set Identifier (CCSID). Use the default (*DFT) value unless you know that you need a specific CCSID. Press the Enter key. The FTP client will display messages that indicate a successful connection with the remote system.

5. To change the file transfer type, do the following:
   a. To switch to EBCDIC, enter **EBCDIC** and press Enter before you transfer the file.
   b. To switch to BINARY, enter **BINARY** and press Enter before you transfer the file.
   c. To switch back to the default type, ASCII, enter **ASCII** and press Enter before you transfer the file.

6. Now you are ready to transfer files:
   a. Enter **CD** and the name of the directory. Press Enter.
   b. Do one of the following:
      - To transfer a file from the server system to the client system, enter **GET** followed by the name of the file:
        
        GET myfile.txt
      - To send a file that is on the client system to the server system, enter **PUT** followed by the name of the file:
        
        PUT myfile.txt

7. Enter the FTP subcommand **QUIT** to end the FTP client session and return to the iSeries 400 command line.

**File transfer types**

Before you begin to transfer files, you must choose the appropriate file transfer type. You can use the default type, ASCII, or specify a different type. ASCII is the Internet standard for character encoding. EBCDIC is the standard for iSeries 400. Select the appropriate type according to the following:

- Use **ASCII (Change File Type to ASCII)** on page 42 for transfers of files that only contain text ("text-only" files).
- Use **EBCDIC (Change File Type to EBCDIC)** on page 46 to transfer EBCDIC data between systems that both support EBCDIC. This will avoid the need to convert data between EBCDIC and ASCII on both systems.
- Use **BINARY (Set Transfer Type to Image)** on page 43 for transfers of non-text files, such as binary numeric data, graphics files, and iSeries 400 save files.

After you have chosen a file transfer format, you are ready to Transfer a file with FTP.

**FTP as batch job**

In addition to running the FTP client interactively, you can run the FTP client in an unattended mode. Two examples of this method are included in this section: a simple example and a complex example.

**Batch FTP: A simple example**

The following is a simple example of a batch file transfer that involves the successful transfer of one file from a remote system.

The components are as follows:

- A CL program
- An input file of FTP commands
• An output file of FTP messages

The CL Program

******************************************************************************

ITSOLIB1/QCLSRC BATCHFTP:

--------------------------------

PGM
OVRDBF FILE(INPUT) TOFILE(ITSOLIB1/QCLSRC) MBR(FTPCMDS)
OVRDBF FILE(OUTPUT) TOFILE(ITSOLIB1/QCLSRC) MBR(OUT)
FTP
RMTSYS(SYSxxx)
ENDPGM

******************************************************************************

The BATCHFTP program overrides the INPUT parameter to the source physical file ITSOLIB1/QCLSRC MBR(FTPCMDS). The output is sent to MBR(OUT).

The Input Commands File

******************************************************************************

ITSOLIB1/QCLSRC FTPCMDS:

---------------------

ITSO ITSO
CD ITSOLIB1
SYSCMD CHGCURLIB ITSOLIB2
GET QCLSRC.BATCHFTP QCLSRC.BATCHFTP (REPLACE
QUIT

******************************************************************************

The FTP subcommands required are shown in the FTPCMDS file.

The Output Messages File

******************************************************************************

FTP Output Redirected to a File
FTP Input from Overridden File
Connecting to host name SYSxxx
at address x.xxx.xx.xxx using port 21.
220 Connection will close if idle more than 5 minutes.
Enter login ID (itso):
> ITSO ITSO
331 Enter password.
230 ITSO logged on.
OS/400 is the remote operating system. The TCP/IP version is "V3R1M0".
250 Now using naming format "O".
257 "QGPL" is current library.
Enter an FTP subcommand.
> CD ITSOLIB1
Enter an FTP subcommand.
250 Current library changed to ITSOLIB1.
> SYSCMD CHGCURLIB ITSOLIB2
Enter an FTP subcommand.
> GET QCLSRC.BATCHFTP QCLSRC.BATCHFTP (REPLACE
200 PORT subcommand request successful.
150 Retrieving member BATCHFTP in file QCLSRC in library ITSOLIB1.
250 File transfer completed successfully.
147 bytes transferred in 0.487 seconds. Transfer rate 0.302 KB/sec.
Enter an FTP subcommand.
> QUIT
221 QUIT subcommand received.

******************************************************************************

The output file is shown. It is a straightforward matter to write a program to process this file and display an error message on QSYSOPR if there are any error messages. FTP error messages have numbers that start with a 4 or 5.
Batch FTP: A Complex Example
The following example shows how to retrieve files from several remote hosts to a central iSeries 400 in batch mode:

User GWIL on iSeries 400 SYSNAM03 wants to:
1. Retrieve files from hosts SYSNAMRS (RS/6000) and MVAX (VAX).
2. After retrieving the file from SYSNAMRS, the file should be transferred to SYSNAM02 (another iSeries 400) using FTP.
3. From there the file is to be sent using SNA to iSeries 400 SYSNAM14.

Create a CL Program to Start FTP
1. As we have seen in the previous example, FTP uses the display station for command INPUT and message OUTPUT, and this needs to be overridden for use in batch mode. We use the OVRDBF command to overwrite these files with the ones to be used in batch:
   
   OVRDBF FILE(INPUT) TOFILE(GERRYLIB/QCLSRC) MBR(FTPCMDS)
   OVRDBF FILE(OUTPUT) TOFILE(GERRYLIB/QCLSRC) MBR(FTPLOG)

2. A host name or an internet address is a required parameter for the STRTCPFTP command that is included in the CL program file. However, if one wants to specify the remote systems in the input commands file instead of the CL program file, then a dummy host name must be specified for the STRTCPFTP command to satisfy the required syntax. This dummy name may be a fictitious host name.
or a real host name. If it is a real name, then the first entry in the input commands file must be a user
ID and a password, and the second entry must be the CLOSE subcommand. If it is not a real host
name, then these entries are not required, and the first entry should be an OPEN subcommand to
connect to the desired server system.

FTP RMTSYS(LOOPBACK)

FTP processes the input file and writes messages to the output file (FTPLOG).

3. After the FTP application ends, delete the overrides:

DLTOVR FILE(INPUT OUTPUT)

The CL program for batch FTP will look like the following example on system SYSNAM01:

```
| Column s...: 1 7 1 B r owse GERRYLIB/QCLSRC |
| | |
| | SEU==> FTPBATCH |
| | FMT ** ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 |
| | *************** Beginning of data ***************************************|
| 0001.00 PGM |
| 0002.00 OVRDBF FILE(INPUT) TOFILE(GERRYLIB/QCLSRC) + |
| 0003.00 MBR(FTPCMDS) |
| 0004.00 OVRDBF FILE(OUTPUT) TOFILE(GERRYLIB/QCLSRC) + |
| 0005.00 MBR(FTPLOG) |
| 0006.00 FTP RMTSYS(LOOPBACK) /* (FTP CL Program) */ |
| 0007.00 DLTOVR FILE(INPUT OUTPUT) |
| 0008.00 ENDPGM |
| | *************** End of data ****************************************************|
| | |
| | F3=Exit F5=Refresh F9=Retrieve F10=Cursor F12=Cancel |
| | F16=Repeat find F24=More keys |
```

Figure 7-30. CL Program FTPBATCH for Batch FTP

Create the FTP Input File (FTCPDMS)

This file has to contain all the FTP client subcommands necessary to connect and log on to the server, set
up for and do the file transfers, close the server connection, and end the client session. The example in
below shows the subcommands used for transferring files to two different remote systems.

```
| Column s...: 1 7 1 B r owse GERRYLIB/QCLSRC |
| | |
| | SEU==> FTPCMDS |
| | FMT ** ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 |
| | *************** Beginning of data ***************************************|
| 0001.00 gwil **** |
| 0002.00 close |
| 0003.00 open sysnamrs |
| 0004.00 user root root |
| 0005.00 ascii |
| 0006.00 syscmd dltf file(gerrylib/rs6) |
| 0007.00 get /Itsotest gerrylib/rs6.rs6 |
| 0008.00 close |
| 0009.00 open mvax |
| 0010.00 user tester tester |
| 0011.00 get screen1.file gerrylib/vax.vax  (replace |
| 0012.00 close |
```
The explanation for the FTP client subcommands shown above, follows. The line numbers on the display correspond to the numbers that follow.

0001 User ID and password for dummy connection within client iSeries 400 SYSNAM03.
0002 Close dummy connection in iSeries 400 SYSNAM03.
0003 Open control connection to RISC System/6000 SYSNAMRS.
0004 USER subcommand with user ID and password for SYSNAMRS.

**Note:** When running FTP in batch mode, the USER subcommand must follow an OPEN subcommand. Both the logon user ID and password parameters for the USER subcommand should be provided. This is different when operating FTP interactively online. When FTP is run interactively online, then the client will automatically initiate a USER subcommand and prompt you for a logon ID. There is no automatic USER subcommand when running FTP in batch mode.

0005 Transfer ASCII data (will be converted on iSeries 400 to/from EBCDIC).
0006 CL command to be run on client iSeries 400: delete file. Instead parameter (REPLACE could be used with the next statement.
0007 Retrieve file from RISC System/6000 system
0008 Close control connection to RISC System/6000 SYSNAMRS.
0009 Open connection to VAX MVAX.
0010 USER subcommand with user ID and password for MVAX.
0011 Retrieve file from VAX replacing existing iSeries 400 file.
0012 Close control connection to VAX MVAX.
0013 Open control connection to remote iSeries 400 SYSNAM02.
0014 USER subcommand with user ID and password for SYSNAM02.
0015 Transfer EBCDIC data (as it is from iSeries 400 to iSeries 400).
0016 Send iSeries 400 file to iSeries 400 SYSNAM02 with TCP/IP.
0017 Send this file from server iSeries 400 SYSNAM03 to remote iSeries 400 SYSNAM14 through SNA network.
0018 Close control connection to iSeries 400 SYSNAM02.
0019 End FTP application.

**Create CL Program for Submitting the FTPBATCH Job**

To schedule the file transfers and run them unattended, create a CL program that submits the FTPBATCH job. In the following example, the file transfers are supposed to run the next Friday, 17:00 hour, in unattended mode.
### Check the FTP Output File for Errors

While running at the scheduled time, FTP creates the data in file member FTPLOG shown above. The data in file member FTPLOG corresponds to original statements found both examples.

<table>
<thead>
<tr>
<th>Columns . . .</th>
<th>1 71</th>
<th>Browse</th>
<th>GERRYLIB/QCLSRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET</td>
<td>FTPSUBMIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMT  **</td>
<td>...+1 ...+2 ...+3 ...+4 ...+5 ...+6 ...+7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

********** Beginning of data ****************************************

0001.00 PGM

0002.00 SBMJOB CMD(CALL PGM(GERRYLIB/FTPBATCH)) +

0003.00 JOB(FTPFRIDAY) OUTQ(QUSRYSYS/GERRYQ) +

0004.00 SCDDATE(*FRI) SCDDTIME(170000) /* FTP for +

0005.00 Friday, 5:00 in the afternoon */

0006.00 ENDPGM

********** End of data ****************************************

F3=Exit  F5=Refresh  F9=Retrieve  F10=Cursor  F12=Cancel

F16=Repeat find  F24=More keys


---

**FTP Output (FTPLOG) After Running FTPBATCH Program (Part 1)**

- Connecting to host name LOOPBACK at address 127.0.0.1 using port 21.
- 220-QTCP at localhost.
- 220 Connection will close if idle more than 5 minutes.
- Enter login ID (gwil):

  >>>GWIL ****

  331 Enter password.

  230 GWIL logged on.

  OS/400 is the remote operating system. The TCP/IP version is "V4R2M0".

  250 Now using naming format "0".

  257 "QGPL" is current library.

  Enter an FTP subcommand.

  > CLOSE

  221 QUIT subcommand received.

  Enter an FTP subcommand.

  > OPEN SYSNAMRS

  Connecting to host name SYSNAMRS at address 9.4.73.198 using port 21.


  Enter an FTP subcommand.

---

<table>
<thead>
<tr>
<th>Columns . . .</th>
<th>1 71</th>
<th>Browse</th>
<th>GERRYLIB/QCLSRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET</td>
<td>FTPSUBMIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMT  **</td>
<td>...+1 ...+2 ...+3 ...+4 ...+5 ...+6 ...+7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

********** Beginning of data ****************************************

0001.00 PGM

0002.00 SBMJOB CMD(CALL PGM(GERRYLIB/FTPBATCH)) +

0003.00 JOB(FTPFRIDAY) OUTQ(QUSRYSYS/GERRYQ) +

0004.00 SCDDATE(*FRI) SCDDTIME(170000) /* FTP for +

0005.00 Friday, 5:00 in the afternoon */

0006.00 ENDPGM

********** End of data ****************************************

F3=Exit  F5=Refresh  F9=Retrieve  F10=Cursor  F12=Cancel

F16=Repeat find  F24=More keys


---

**FTP Output (FTPLOG) After Running FTPBATCH Program (Part 1)**

- Connecting to host name LOOPBACK at address 127.0.0.1 using port 21.
- 220-QTCP at localhost.
- 220 Connection will close if idle more than 5 minutes.
- Enter login ID (gwil):

  >>>GWIL ****

  331 Enter password.

  230 GWIL logged on.

  OS/400 is the remote operating system. The TCP/IP version is "V4R2M0".

  250 Now using naming format "0".

  257 "QGPL" is current library.

  Enter an FTP subcommand.

  > CLOSE

  221 QUIT subcommand received.

  Enter an FTP subcommand.

  > OPEN SYSNAMRS

  Connecting to host name SYSNAMRS at address 9.4.73.198 using port 21.


  Enter an FTP subcommand.
> ASCII
200 Type set to A; form set to N.
Enter an FTP subcommand.

> SYSCMD DLTF FILE(GERRYLIB/RS6)
Enter an FTP subcommand.

> GET /Itsotest GERRYLIB/RS6/RS7
200 PORT command successful.
150 Opening data connection for /Itsotest (467 bytes).
226 Transfer complete.
467 bytes transferred in 2.845 seconds. Transfer rate 0.167 KB/sec.
Enter an FTP subcommand.

FTP Output (FTPLOG) after Running FTPBATCH Program (Part 2)

> CLOSE
221 Goodbye.
Enter an FTP subcommand.

> OPEN MVAX
Connecting to host system mvax at address 9.4.6.252 using port 21.
220 FTP Service Ready

> USER TESTER ******
331 User name TESTER received, please send password
230 TESTER logged in, directory $DISK1:[TESTER]
Enter an FTP subcommand.

GET SCREEN1.FILE GERRYLIB/VAX.VAX (REPLACE
200 PORT Command OK.
125 ASCII transfer started for $DISK1:[TESTER SCREEN1.FILE;1(266586 bytes)
226 File transfer completed ok.
265037 bytes transferred in 8.635 seconds. Transfer rate 30.694 KB/sec.
Enter an FTP subcommand.

> CLOSE
221 Goodbye.
Enter an FTP subcommand.

OPEN SYSNAM02
Connecting to host system SYSNAM02 at address 9.4.73.250 using port 21.
220 Connection will close if idle more than 5 minutes.

FTP Output (FTPLOG) after Running FTPBATCH Program (Part 3)

> USER GWIL ****
331 Enter password.
230 GWIL logged on.

OS/400 is the remote operating system. The TCP/IP version is "V4R2M0".
250 Now using naming format "0".
257 "QGPL" is current library.
Enter an FTP subcommand.
FTP client subcommands

File Transfer Protocol (FTP) subcommands instruct the FTP client to transfer files from one computer to another. With FTP client subcommands, you can establish a connection with a remote FTP server, navigate libraries and directories, create and delete files, and transfer files.

You can access descriptions about client subcommands and their syntax from topics in this section.

Related topics:
- FTP client syntax conventions
- FTP server subcommands
- FTP server reply status messages
- Transferring files with FTP

Sample Procedure: A sample REXX procedure and a sample physical file member are shipped as part of the TCP/IP product. File QATMPINC in library QTCP includes the following two members:
- BATCHFTP that contains REXX source code to specify the input and output batch files, and start FTP.
- BFTPFILE that contains the subcommands and data required for logon and running FTP.
iSeries 400 FTP client supports these subcommands. This table identifies client subcommands, the accepted abbreviations, and each subcommand’s function.

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>What It Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Describes How to Use FTP</td>
</tr>
<tr>
<td>ACCT</td>
<td>Sends a User’s Account Information to a Remote System</td>
</tr>
<tr>
<td>APPEND</td>
<td>Adds a Local File Member to File on a Remote System</td>
</tr>
<tr>
<td>ASCII</td>
<td>Sets the File Transfer Type to ASCII Format</td>
</tr>
<tr>
<td>BINARY</td>
<td>Sets the File Transfer Type to BINARY format</td>
</tr>
<tr>
<td>CD</td>
<td>Changes the Working Directory on Remote System</td>
</tr>
<tr>
<td>CDUP</td>
<td>Changes to the Parent Directory on Remote System</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Ends a Session with the Remote System</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Turns Debugging On or Off</td>
</tr>
<tr>
<td>DELETE</td>
<td>Deletes a File on the Remote System</td>
</tr>
<tr>
<td>DIR</td>
<td>Displays Directories and Files on the Remote System</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>Sets the File Transfer Type to EBCDIC Format</td>
</tr>
<tr>
<td>GET</td>
<td>Copies a File from the Remote to a Local System</td>
</tr>
<tr>
<td>HELP</td>
<td>Gets Information about FTP Client Subcommands</td>
</tr>
<tr>
<td>Subcommand</td>
<td>What It Does</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LCD</td>
<td>Changes the Working Directory on Local System</td>
</tr>
<tr>
<td>LOCSDL</td>
<td>Specifies Local Site Information</td>
</tr>
<tr>
<td>LOCSTAT</td>
<td>Displays Local Status Information</td>
</tr>
<tr>
<td>LPWD</td>
<td>Displays the Working Directory on Local System</td>
</tr>
<tr>
<td>LS</td>
<td>Lists the Names of Files in a File Set on Remote System</td>
</tr>
<tr>
<td>LTYPE</td>
<td>Specifies the File Transfer Type on the Local System</td>
</tr>
<tr>
<td>MDELETE</td>
<td>Deletes Multiple Files on the Server System</td>
</tr>
<tr>
<td>MGET</td>
<td>Copies File or Files from the Remote System</td>
</tr>
<tr>
<td>MKDIR</td>
<td>Creates a Directory or Subdirectory</td>
</tr>
<tr>
<td>MODE</td>
<td>Specifies a Data Format for File Transfer</td>
</tr>
<tr>
<td>MPUT</td>
<td>Sends Local File or Files to the Remote System</td>
</tr>
<tr>
<td>NAMEFMT</td>
<td>Specifies a File Naming Format to Use</td>
</tr>
<tr>
<td>NOOP</td>
<td>Checks for a Response</td>
</tr>
<tr>
<td>NULLFLDS</td>
<td>Allows for NULL Fields</td>
</tr>
<tr>
<td>OPEN</td>
<td>Connects to an FTP Server</td>
</tr>
<tr>
<td>PASS</td>
<td>Sends a User's Password</td>
</tr>
<tr>
<td>Subcommand</td>
<td>What It Does</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT (Copy a File Member from the Local System to a File on a Remote System) on page 55</td>
</tr>
<tr>
<td>PWD</td>
<td>PWD (Display Current Directory, Folder or Library) on page 56</td>
</tr>
<tr>
<td>QUIT</td>
<td>Ends an FTP Session</td>
</tr>
<tr>
<td>QUOTE</td>
<td>QUOTE (Send a Subcommand to an FTP Server) on page 57</td>
</tr>
<tr>
<td>REINITIALIZE</td>
<td>REINITIALIZE (Reinitialize Session between Systems) on page 57</td>
</tr>
<tr>
<td>RENAME</td>
<td>RENAME (Rename a File on a Remote System) on page 58</td>
</tr>
<tr>
<td>RESET</td>
<td>RESET (Reset) on page 58</td>
</tr>
<tr>
<td>RMDIR</td>
<td>RMDIR (Remove Directory) on page 59</td>
</tr>
<tr>
<td>SENDPASV</td>
<td>SENDPASV (Specify Whether to send a PASV Subcommand) on page 59</td>
</tr>
<tr>
<td>SENDPORT</td>
<td>SENDPORT (Specify Whether to Sends a PORT Subcommand) on page 60</td>
</tr>
<tr>
<td>SENDSITE</td>
<td>SENDSITE (Specify Whether to Send a SITE Subcommand) on page 60</td>
</tr>
<tr>
<td>SITE</td>
<td>SITE (Send Information Used by a Remote System) on page 61</td>
</tr>
<tr>
<td>STATUS</td>
<td>STATUS (Retrieve Status Information from a Remote System) on page 61</td>
</tr>
<tr>
<td>STRUCT</td>
<td>STRUCT (Specify File Structure) on page 62</td>
</tr>
<tr>
<td>Subcommand</td>
<td>What It Does</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>SUNIQUE</td>
<td>Controls File Replacement</td>
</tr>
<tr>
<td>SYSCMD</td>
<td>Runs a CL Command on a Local System Without Quitting FTP</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>Displays the OS on the Remote System</td>
</tr>
<tr>
<td>TYPE</td>
<td>Specifies the File Transfer Type</td>
</tr>
<tr>
<td>USER</td>
<td>Sends a User ID to a Remote System</td>
</tr>
<tr>
<td>VERBOSE</td>
<td>Controls the Display of FTP Server Replies</td>
</tr>
</tbody>
</table>

**ACCT (Send Account Information)**

*FTP Client Subcommand*

Some systems require account information to enable certain system functions. The remote system prompts you for such information. To send account information, use the Account (ACCT) FTP client subcommand:

```
ACCT account-information
```

**account-information**

A string that identifies the user’s account. Account information can take the form of a password that the host system uses to grant privileges. This password is not your user password, but rather it is a password on the remote system.

For example, TCP/IP on the IBM Virtual Machine (VM) Operating System may require a password for read and write access to minidisks. Use the ACCT subcommand to supply a password for the minidisk of the current directory. If the remote system is an iSeries 400 system, the ACCT subcommand performs no operation.

**Go To:**
- FTP client subcommands [FTP client subcommands” on page 37]
- FTP client syntax conventions [FTP client syntax conventions” on page 66]

**APPEND (Append a Local File Member to a Remote File)**

*FTP Client Subcommand*

To add a local file member, document, or other file system file to a remote file, use the APPEND FTP client subcommand in this format:

```
APPend localfile [remotefile]
```
localfile
The name of the local file member, document, or other iSeries 400 file. The name of the hierarchical file system (HFS) file added to a directory on the remote system. For information on file naming, see NAMEFMT (Select File Naming Format).

remotefile
The file on the remote system. If you do not enter a remote file, the FTP client creates a default name. For information on how FTP creates default names, see Default file names for client transfer subcommands on page 67.

If the remote file does not exist on the server, the FTP server creates it.

To add a file on the remote system, you must have write privileges to it. You may have to supply the appropriate account information by using the ACCT subcommand (see ACCT (Send Account Information) on page 41).

The default file copy mode is stream. You may need to change this by using the MODE subcommand. In the case of fixed-record format in the remote file, the server preserves the file format and record length of the remote file. Records from the local file member shorten or include blanks when necessary.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

ASCII (Change File Type to ASCII)
FTP Client Subcommand

AScii

Use the ASCII transfer type when transferring text files to or from an ASCII system that does not support EBCDIC representation. ASCII is the default transfer type. The server does not associate a vertical format control to the file. ASCII only supports the default format NON PRINT.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

BINARY (Set Transfer Type to Image)
FTP Client Subcommand

Binary

If you are transferring binary data to an existing iSeries 400 file, the record length is the record length of the existing iSeries 400 file. For example the existing file size should accommodate the new data. If the file does not exist on an iSeries 400 system, FTP chooses a record length for you.

Certain files, such as save files, require binary image transfer. If TYPE is not binary when attempting to transfer such files, you receive a message tells you to use binary.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66
CD (Change Working Directory or Library)
FTP Client Subcommand

Use the Change Directory (CD) FTP client subcommand to change the working directory, library, or file group on the remote system:

CD directory

directory
The name of a file directory, library, or other system-dependent file-group designator on the remote system.

If the remote system is an iSeries 400 system, this subcommand changes the current library or directory. To find out what directories are on the remote system, use the Directory (DIR) subcommand to get a listing.

Use the DIR subcommand with caution. See DIR (List Directory Entries, Libraries, or Files) on page 45 for further details and advice.

Note: When using the subcommand CD (or LCD) to change from one iSeries 400 file system to another, you must specify the root directory of the file system that contains the new current directory.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

CLOSE (End an FTP Session with the Remote System)
FTP Client Subcommand

To end your session with the remote system and keep FTP active on your local iSeries 400 system, use the CLOSE FTP client subcommand:

Close

The CLOSE subcommand allows you to remain in the FTP environment to open another FTP session on another system. Use the OPEN subcommand to establish a new connection with the same remote system or another remote system. Use the QUIT subcommand to end FTP service and return to the iSeries 400 environment from which FTP was started.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

DEBUG (Create Client Trace and Control Display of Server Subcommands Sent to Remote System)

Note: You should only use the FTP client trace to report software problems to IBM. System performance may be adversely affected by this function.

This capability is available in release V4R4 and above of OS/400.
FTP Client Subcommand

To produce an FTP client trace or display, use the DEBUG FTP client subcommand. The DEBUG subcommand toggles the debugging mode. If the client specifies an optional debug-value, it will use it to set the debugging level. When debugging is on, the client displays with the string ‘>>>’. You must set the debug-value to 100 to produce an FTP client trace.

DEBug [debug value]

debug value
If the debug-value is 0, debugging is off. If the debug-value is a positive integer, debugging is on.
If you don’t specify a value, the debug value toggles from zero to one or from a positive integer to zero.
100 Initiate an FTP client trace. The client continues running the trace until the DEBUG is off or until the server ends the FTP client. When the server ends the trace, there may be a significant delay while it formats the trace data.

To initiate a trace immediately when the FTP client starts, you need to create the QTMFTPD100 data area in the QTEMP library by using this command:

CRTDTAARA DTAARA(QTEMP/QTMFTPD100) TYPE(*LGL) AUT(*USE)

If the QTMFTPD100 data area exists, then it will set the debug value to 100 and start an FTP client trace. The purpose of this capability is to enable the FTP client debug traces in those situations when an FTP client trace cannot start with the DEBUG 100 subcommand.

Go To:
• FTP client subcommands
• FTP client syntax conventions

DEBUG (Change Client Time-Out Limit Values)
FTP Client Subcommand

To change the client time-out limits when the default time-out values are not long enough for a data transfer to complete successfully, use the DEBUG subcommand. You should only need to change these values in situations where network traffic or other conditions cause transfer times to become very large.

To change the FTP client time-out values, use the DEBUG FTP client subcommand:

DEBug T1 | T2 [ value ]

T1 Change or display the FTP client time-out limit for reading server replies. If the FTP client does not receive an expected server reply within this time limit, the client will close the control connection to the server.

T2 Change or display the FTP client time-out limit for transferring data. If the FTP client does not receive an expected data connection response within this time limit, the client will close the data connection to the server.

value The time-out limit in seconds. This value must be a positive number greater than zero. When you omit this value, the client displays the current value of the time-out limit.

For example:

DEBug T1 900
This value sets the client time-out value for server replies to 900 seconds.

Go To:
- FTP client subcommands [FTP client subcommands] on page 37
- FTP client syntax conventions [FTP client syntax conventions] on page 66

DELETE (Delete a File on a Remote System)
FTP Client Subcommand

Use the DELETE subcommand to delete a file or database file member on a remote system. The DELETE FTP client subcommand format is:

```
DELete remotefile
```

remotefile
The file you want to delete on the remote system. For information on how to specify the file if the remote system is an iSeries 400 business computing system, see NAMEFMT (Select File Naming Format) on page 54.

The remote system may prompt you for authorization to delete a file. Use the ACCT (Send Account Information) subcommand to respond to that request.

See also MDELETE (Delete Multiple Files on a Remote System) on page 51.

Go To:
- FTP client subcommands [FTP client subcommands] on page 37
- FTP client syntax conventions [FTP client syntax conventions] on page 66

DIR (List Directory Entries, Libraries, or Files)
FTP Client Subcommand

The DIR FTP client subcommand displays libraries and their contents or the remote system’s list of directories and directory entries. Use the Directory (DIR) subcommand in this format:

```
Dir [name] [(Disk]
```

name
The name of the directory or library. The default is the entire current directory or library. To make a library or directory current, use the Change Working Directory (CD) subcommand. How you specify a set of remote files depends on the system. Most systems allow a generic asterisk, *. If the remote system is an iSeries 400 system, for example,

```
DIR MYLIB/MYFILE.*
```
produces a list of all members of MYFILE in library MYLIB.

There are two possible file name formats you can use. The example shown here uses NAMEFMT 0. For information about FTP file naming, see NAMEFMT (Select File Naming Format).

( Disk
Stores the results of the DIR subcommand in the file *CURLIB/DIROUTPUT.DIROUTPUT, instead of showing the results on the display.

If the remote system is an iSeries 400 system, the information includes:
- For database files, the *FILE objects, and members.
For hierarchical file system (HFS) files:
- All document library services (QDLS) folders and their contents, which could be other folders or documents.
- All optical volumes (QOPT) and their contents, which could be directories or files.

Use the DIR subcommand with caution. If you enter the DIR subcommand without any parameters, the server produces a listing of all the current directory files. This may be a much longer list than you want.

To get a list of the file names in a directory, use the List (LS) subcommand (see LS (List Remote File Names) on page 50).

EBCDIC (Change File Type to EBCDIC)
FTP Client Subcommand

The EBCDIC transfer type is useful when transferring files to or from another EBCDIC system. This is due to the fact that it avoids the need to convert between ASCII and EBCDIC on both systems.

GET (Copy a File from a Remote System to the Local System)
FTP Client Subcommand

remotefile
The file that you want to retrieve from the remote system.

localfile
The local file member, document, or other file you want to create. If you do not specify a local file name, the FTP client provides a default name. For information about the default names, see Default file names for client transfer subcommands.

( Replace
Writes over the localfile if it already exists. The server will not overwrite the localfile unless you specify (Replace.

The file system in which the file resides determines which file name format you use with the GET subcommand.
- If the file does not reside in the library file system (QSYS.LIB), you must use the GET subcommand in name format (NAMEFMT) 1:
  GET /QDLS/QIWSOS2/PCSMENU.EXE
- If the file resides in the library file system, use the GET subcommand and the name format (NAMEFMT) set to 0:
  GET YOURLIB/YOURFILE.YOURMBR (REPLACE
Assuming the remote server is an iSeries 400 system, this command gets the YOURMBR of YOURFILE in YOURLIB and places it in YOURMBR of YOURFILE in your current directory on your local system. For more information on defining the current directory, see LCD (Change Working Library or Directory on Local System) on page 48.

Note: If the remote file name requires apostrophes as part of the file name, then enclose the file name within two more sets of apostrophes. The following example gets "MEMBER.ONE" from the remote host. See Enclosing subcommand parameters on page 67 for more information.

GET LIBRARY/FILE.MEMBER 'MEMBER.ONE'

Go To:
- FTP client subcommands on page 37
- MGET (Copy Multiple Files from a Remote System to the Local System) on page 51
- PUT (Copy a File Member from the Local System to a File on a Remote System) on page 53
- MPUT (Send Multiple File Members from the Local System to a Remote System) on page 53
- FTP client syntax conventions on page 66

HELP (Getting Help for FTP Subcommands)
The HELP subcommand provides information about the FTP subcommands that the local system and the remote system uses.

Help for FTP Client Subcommands
To get information about FTP subcommands used by the local system, use the HELP subcommand in this format:

Help [ * | ALL | subcommand ]

* or ALL
Displays a list of the FTP client subcommands.

subcommand
Provides detailed help for the specified client subcommand. For example, HELP GET tells you how to transfer a file from a remote system to your local system. You may abbreviate the subcommand to a meaningful prefix.

If you use the HELP subcommand without a parameter, you see a list of subcommands and a general description of the help information available. Context-sensitive help is available by positioning the cursor over a command on the help display and then pressing the Enter key.

To get the list of local subcommands on an iSeries 400 system, enter:
HELP

Help information can be obtained with the ? subcommand.

Help for FTP Server Subcommands
To obtain help for FTP subcommands on the remote system, use the HELP subcommand in this format:
Help SERVER [subcommand]

SERVER
Gives the help the remote system offers for FTP server subcommands. This is similar to using QUOTE with the HELP parameter. QUOTE HELP lists the FTP subcommands supported by the remote system.

subcommand
The name of the server subcommand that you want the information. For example, HELP SERVER STOR will request the server to provide help on the STOR subcommand.

Note: RHELP is a synonym for HELP SERVER. For example, HELP SERVER SITE and RHELP SITE are equivalent.

For additional information, see QUOTE (Send a Subcommand to an FTP Server) on page 57.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

LCD (Change Working Library or Directory on Local System)
FTP Client Subcommand

LCd pathname

pathname
The name of a library, folder, or directory on the local system.

Notes:
1. The LCD subcommand does not change the current library entry of the library list.
2. When using the subcommand CD (or LCD) to change from one file system to another file system, you must specify the “root” directory For example, /QDLS or /QOPT.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

LOCSITE (Specify Local Site Information)
FTP Client Subcommand

To specify information that is used by the FTP client to provide services specific to the client system, use the LOCSITE FTP client subcommand:

LOCSITE [ parameters ]

iSeries 400 FTP client supports these parameters and parameter options for the LOCSITE subcommand:

CRTCCSID *CALC
New database files created during ASCII file transfers use the related default EBCDIC CCSID of the ASCII file transfer CCSID. This is the default value.
CRTCCSID *USER
New database files created during ASCII file transfers use the current job CCSID. However, if this
cCSID is 65535, the default CCSID determined by the language id in the current job specification.

CRTCCSID *SYSVAL
New database files created during ASCII file transfers use the CCSID that the QCCSID system
value specifies.

CRTCCSID [CCSID-number]
Specify the CCSID you want to use when creating database files on the client during ASCII file
transfers. The server validates this value.

CRTCCSID
Display a message that contains the current FTP client CRTCCSID setting.

TRIM 0
Set Trim option to OFF. The server sends trailing blanks of database records.

TRIM 1
Set Trim option to ON. The server does not send trailing blanks of database records when
transferring database files that use file structure and stream mode. This is the default.

TRIM 2
Set Trim option so the server does not send trailing blanks of database records for all transfers,
including record structure and block mode.

TRIM
Display a message that contains the current setting of the FTP client TRIM option.

Notes:
1. Prior to the availability of this subcommand, trailing blanks of QSYS.LIB file system records were
   always removed before transferring the file to the server system.
2. TRIM settings do not apply to TYPE I (binary) file transfers. Blanks are never trimmed for TYPE I file
   transfers, regardless of the TRIM setting.

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

LOCSTAT (Display Local Status Information)
FTP Client Subcommand

Displays Local status information, including:
• The current setting of the SENDSITE subcommand
• The current setting of the SENDPORT subcommand
• Remote system name, port number, and logon status
• Data type and transfer mode
• Name format value for both the client and the server
• Setting for the VERBOSE mode
• Setting for the DEBUG mode

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
FTP client syntax conventions on page 66

**LS (List Remote File Names)**

FTP Client Subcommand

**LS [name] [[Disk]**

**name**: The remote directory, file, or library that you want to list. If the remote system is an iSeries 400 business computing system, the server lists the file names and its members. The default is to list the entire current directory, library, or folder. To change the current directory, library, or folder, use the **CD (Change Working Directory or Library)** subcommand (see **CD (Change Working Directory or Library)** on page 43). The remote file specification is system dependent.

**(Disk)** Stores the results of the LS subcommand in the file *CURLIB/LSOUTPUT.LSOUTPUT*, instead of showing the results on the display. Each time you specify the (Disk parameter with the same *CURLIB, the server changes the contents of the LSOUTPUT.LSOUTPUT member file.

**Note:** If the FTP server returns a negative reply code (550), then there will be no LSOUTPUT member. If the FTP server returns a positive reply code (150) without any file names, then an LSOUTPUT member with no records will result.

The LS subcommand lists the file names only. To get a list of complete directory entries with additional information about the files see **DIR (List Directory Entries, Libraries, or Files)**.

**Go To:**

- FTP client subcommands on page 37
- FTP client syntax conventions on page 66

**LTYPE (Local Type)**

FTP Client Subcommand

To specify the file transfer type or the representation in which the transfer is to take place on the local system, use the LTYPE FTP client subcommand in this format:

**LType C ccsid#**

**C**: The CCSID type. Code this value as C.

**ccsid#**: The CCSID value. Code this value as a CCSID number 1-65533.

**Note:** The LTYPE subcommand is similar to the TYPE subcommand (see **TYPE (Specify File Transfer Type)**). The LTYPE subcommand changes only the representation type on the client side. The TYPE subcommand changes the representation type on both the client and the server.

**Go To:**

- FTP client subcommands on page 37
- FTP client syntax conventions on page 66
MDELETE (Delete Multiple Files on a Remote System)
FTP Client Subcommand

MDelete \(remotefile\ [remotefile...]\)

\textbf{remotefile} \(\textit{á}\)
The file or files on the server system you want to delete.

\textbf{Note:}
When the remotefile is an iSeries 400 QSYS.LIB file, then the server deletes all members of the iSeries 400 physical file. The file itself remains.

This is a library file system example in NAMEFMT 0:
MDELETE MYLIB/FILE1.MBRA YOURLIB/FILE2.MBRB

This deletes member MBRA in file FILE1 in library MYLIB and member MBRB in file FILE2 in library YOURLIB on a remote iSeries 400 system. The same example in NAMEFMT 1:
MDELETE /QSYS.LIB/MYLIB.LIB/FILE1.FILE/MBRA.MBR
/QSYS.LIB/YOURLIB.LIB/FILE2.FILE./MBRB.MBR

This is a document library system example in NAMEFMT 1:
MDELETE /QDLS/QIWSOS2/PCSMENU.EXE /QDLS/PCSDIR/PCSFILE.EXE

This deletes document PCSMENU.EXE in folder QIWSOS2 in the document library services library, and also deletes PCSFILE.EXE in folder PCSDIR in the QDLS library on an iSeries 400 system.

You can use an asterisk (\(\ast\)) to delete the files generically. For example with NAMEFMT 0, if the remote system is an iSeries 400 system, type:
MDELETE MYLIB/MYFILE.\(\ast\)

This example would delete all members of file MYFILE in library MYLIB. Use of the asterisk is only valid at the end of a character string.

\textbf{Go To:}
- FTP client subcommands"FTP client subcommands" on page 37
- FTP client syntax conventions"FTP client syntax conventions" on page 66

MGET (Copy Multiple Files from a Remote System to the Local System)
FTP Client Subcommand

\textbf{How MGET transfers files:}

To copy one or more remote files, use the MGET FTP client subcommand in this format. A separate GET subcommand is executed for each remote file you want transferred. The server creates the name of the corresponding local file automatically as determined by the Default Naming rules.

The MGET FTP client subcommand uses the following process to determine where to put files.
- The MGET subcommand always places files in the current library or directory.
- If the user has issued the LCD subcommand, the server uses this library or directory.
- If the user has not issued the LCD subcommand, the server sets the current directory as follows
  - If the user’s job has a current library set, this library is the current directory for FTP.
  - If the user’s job does not have a current library set, the server uses QGPL as the current directory.
MGet {remotefile
[remotefile...]}
(Replace)

remotefile
The file or files you want to retrieve from the remote system.

(Replace)
Overwrites an existing file on your local system. If the file already exists on your local system and you do not use the Replace option, the existing file is not overwritten. The name of the local file where the remotefile is copied is created automatically.

See the GET subcommand description GET (Copy a File from a Remote System to the Local System) on page 48 for additional information.

You can use an asterisk (*) to copy all members in a file to your current library or directory. For example, if the remote system is an iSeries 400 system,
- MGET MYLIB/MYFILE.* copies all the members of file MYFILE in library MYLIB on the remote system to your current library on the local system.
- MGET /QSYS.LIB/MYLIB/LIB/MYFILE.FILE/*.MBR would be the NAMEFMT 1 version of this command.
- MGET /QOPT/PICTURES/IMAGES/*. copies all the files of directory IMAGES from optical volume PICTURES to your current library (or directory) on the local system.
- MGET TESTFILE.A* copies all members that start with the letter A in file TESTFILE.
- MGET /QDLS/QISSOS2/A* copies all documents that start with the letter A in folder QISSOS2.

Go To:
- FTP client subcommands on page 37
- FTP client syntax conventions on page 66

MKDIR (Make Directory)
FTP Client Subcommand
MKdir pathname

pathname
The name of a file directory, library, or other system-dependent file-group designator on the remote system.

Go To:
- FTP client subcommands on page 37
- FTP client syntax conventions on page 66

MODE (Specify Transmission Mode of Data)
FTP Client Subcommand
MODE [B | S]

B Specifies block mode. In this mode, the server transmits data as a series of data blocks, preceded by one or more header bytes. If you are transferring data in block mode, the type must be EBCDIC.
S  Specifies stream mode. In this mode, the server transmits data as a stream of bytes. You can use any representation type with stream mode.

Notes:
1. Stream mode is the default transfer mode that is used in FTP. Some systems do not support block mode.
2. If you omit the optional parameter, the client displays the present MODE value.

Go To:
• FTP client subcommands
• FTP client syntax conventions

MPUT (Send Multiple File Members from the Local System to a Remote System)
FTP Client Subcommand

To send one or more local files to the remote system, use the MPUT FTP client subcommand. The client executes a separate PUT subcommand for each local file you want transferred. The Default Naming rules create the name of the corresponding remote file.

MPut {localfile [localfile...]}  

localfile  
Specify one or more local library file system file members or other FTP supported file system files you want transferred to the remote system. The client automatically generates the name given to the file on the remote system.

Note:  
If the remote file already exists, the contents are replaced by the contents of localfile unless Store Unique (SUNIQUE) is on (see SUNIQUE (Control Overwriting of Files) on page 62).

For information as to how to specify the file if the remote system is an iSeries 400 system, see NAMEFMT (Select File Naming Format) on page 54. This example uses NAMEFMT 0:

MPUT MYLIB/FILE1.MBR1 MYLIB/FILE1.MBR2  
This sends members MBR1 and MBR2 of file FILE1 in library MYLIB to the remote system.

This example uses NAMEFMT 1:

MPUT /QDLS/QIWSOS2/PCSMENU.EXE /QDLS/QIWSOS2/PCSMENU2.EXE  
This sends document PCSMENU.EXE and document PCSMENU2.EXE from folder QIWSOS2 to the remote system.

You can use an asterisk (*) to send all the members in a file. For example, MPUT MYLIB/FILE.* transfers all the members of file MYFILE in library MYLIB. For additional information, see MGET (Copy Multiple Files from a Remote System to the Local System).

Go To:
• FTP client subcommands
• PUT (Copy a File Member from the Local System to a File on a Remote System)
FTP client syntax conventions

NAMEFMT (Select File Naming Format)
FTP Client Subcommand

To select which file name format to use on the local system and the remote system (if it is an iSeries 400 system), use the NAMEFMT FTP client subcommand:

NAmefmt [ 0 | 1 ]

0 A name format only for library file system database files. The general format is:
   [libname/]filename[.mbrname]

1 A name format for all file systems that FTP supports, including the library file system. You must set the name format to '1' to work with all iSeries 400 file systems.

Library file system files in this name format are:
   [/QSYS.LIB/][libname.LIB/]filename.FILE[/mbrname.MBR]

For save files, you can also use the format:
   /QSYS.LIB/libname.LIB/filename.SAVF

Files in the document library services file system are in this format:
   [/QDLS/][foldername[.ext]/]filename[.ext]

For optical, the format is:
   /QOPT/volname/dirname/filename.ext

Notes:
1. You can set the name format to 0 only when the working directory is a database library.
2. If you specify the NAMEFMT subcommand without a parameter, the client displays the current name format.

See also File systems and naming conventions FTP supports on page 23.

Go To:
• FTP client subcommands on page 37
• FTP client syntax conventions on page 66

NULLFLDS (Allow Transfer of Files with NULL Fields)
FTP Client Subcommand

Use this command to select whether or not to allow transfer of a database files that contain NULL field values on the local system, and the remote system if it is an iSeries 400 system.

NUllflds [ 0 | 1 ]

When you enter a parameter the valid values are:

0 Do not allow transfer of database files that contain NULL fields. This is the default.
1 Allow transfer of database files that contain NULL fields.
Note:

1. Transfer of files that contain NULL fields requires both the client and server to have this setting enabled. The target file must exist prior to the file transfer. Also, the target file must have the same file definition as the source file.

2. Results are not predictable if you transfer a file that contains NULL fields a non-iSeries 400 system, or if the transfer type results in codepage conversion of the data.

3. If you specify the NULLFLDS subcommand without a parameter, the client displays the current setting.

Go To:
- FTP client subcommands
- FTP client syntax conventions

OPEN (Connect to FTP Server on a Remote System)
FTP Client Subcommand

Open systemname [portnumber]

systemname
   The name or Internet address of the remote system.

portnumber
   The port number to use for this session until the server closes the connection. This is optional. If you do not specify a port number, the server chooses one.

Once you have opened a connection to a remote system, you cannot connect to another system until you close the current session.

Go To:
- FTP client subcommands
- FTP client syntax conventions

PASS (Send Your Password)
FTP Client Subcommand

PAss password

password
   A string that specifies your password.

the OPEN and USER subcommands must precede this subcommand. For some systems, this completes your identification for access control. This subcommand is not necessary when the server requests you to type a password when connecting or logging on to the server.

Go To:
- FTP client subcommands
- FTP client syntax conventions

PUT (Copy a File Member from the Local System to a File on a Remote System)
FTP Client Subcommand
**Put** localfile [remotefile]

**localfile**
The name of a local library system file member, save file, document, or other file.

**remotefile**
The name of the delivered file on the remote system. If you do not specify the remote file name, the FTP server provides a default name. For information about default names, see Default file names for client transfer subcommands. If a remote file with the same name already exists, then the server replaces the contents with the contents of the local file unless Store Unique (SUNIQUE) is on (see SUNIQUE (Control Overwriting of Files)).

To send a file to the remote system, you must have a defined current working directory with write privileges.

This example uses the **PUT** subcommand to transfer a file member:

```
PUT MYLIB/MYFILE,MYMBR (NAMEFMT = 0)
```

This sends member MYMBR of file MYFILE in library MYLIB to the remote system.

This example sends the document PCSMENU.EXE of folder QIWSOS2 in the document library services file system to the remote system.

```
PUT /QDLS/QIWSOS2/PCSMENU.EXE (NAMEFMT = 1)
```

**Note:**
If the remote file name requires apostrophes as part of the file name, then you must enclose the file name within two more sets of apostrophes. The following example sends 'MEMBER.ONE' as the file name to the remote host. See Enclosing subcommand parameters on page 67 for more information.

```
PUT LIBRARY/FILE.MEMBER 'MEMBER.ONE'
```

**Go To:**
- FTP client subcommands on page 37
- MPUT (Send Multiple File Members from the Local System to a Remote System) on page 53
- GET (Copy a File from a Remote System to the Local System) on page 46
- MGET (Copy Multiple Files from a Remote System to the Local System) on page 51
- FTP client syntax conventions on page 66

**PWD (Display Current Directory, Folder, or Library)**
**FTP Client Subcommand**

To display the current directory or library of the remote system, use the PWD FTP client subcommand:

**PWd**
If the remote system is an iSeries 400, the server displays your current library or file system directory on the remote system. Also, the server displays the working directory in quotation marks. To change the current library or directory of the remote system, use the Change Working Directory (CD) subcommand.

**Go To:**
- FTP client subcommands [FTP client subcommands](#) on page 37
- FTP client syntax conventions [FTP client syntax conventions](#) on page 66

### QUOTE (Send a Subcommand to an FTP Server)

#### FTP Client Subcommand

<table>
<thead>
<tr>
<th>QUOte string</th>
</tr>
</thead>
</table>

**string** The server subcommand you want sent to and interpreted by the remote FTP server. The FTP server sends the string verbatim to the remote FTP server.

**Notes:**
1. The client requires the QUOTE subcommand to run the special iSeries 400 FTP server subcommand RCMD. For example, to write the server job log to a spooled file, enter this:
   ```
   QUOTE RCMD DSPJOBLOG
   ```
   See RCMD (Send a CL Command to an FTP Server System) [RCMD (Send a CL Command to an FTP Server System)](#) on page 81.
2. iSeries 400 FTP server limits the string to 1000 characters.
3. For the QUOTE subcommand, whatever you enter passes on to the server. For example, if you enter:
   ```
   QUOTE CWD 'SYS1'
   ```
   The server receives
   ```
   CWD 'SYS1'
   ```
   You can get help information from the server by typing this:
   ```
   QUOTE HELP
   ```
   The server sends the HELP subcommand to the remote host, which returns a display of all subcommands it supports. The information displayed varies depending on the type of remote host.

   It should be noted that server subcommands entered with the QUOTE subcommand only affect the server, but similar client subcommands may affect both the client and the server. For example, the REIN client subcommand sends the server a REIN server subcommand plus reinitializes certain client state variables. QUOTE REIN sends only REIN to the server, but does not change any client state variables.

   **CAUTION:** Be careful when using the QUOTE subcommand to directly enter server subcommands so that unintended results do not occur. Typically, use the QUOTE subcommand for special situations that cannot use other client subcommands. An example of this is when one wants to use one of the special iSeries 400 server subcommands like CRTL.

**Go To:**
- FTP client subcommands [FTP client subcommands](#) on page 37
- FTP client syntax conventions [FTP client syntax conventions](#) on page 66

### REINITIALIZE (Reinitialize Session between Systems)

#### FTP Client Subcommand
REInitialize

If the server supports the REINITIALIZE subcommand, the USER session with the server is ended. The server is in the same state as when the connection was established, and the user needs to log on again to continue.

Any file transfers already in progress can complete before the USER session ends.

Go To:
• FTP client subcommands [FTP client subcommands on page 37]
• FTP client syntax conventions [FTP client syntax conventions on page 66]

RENAME (Rename a File on a Remote System)
FTP Client Subcommand

REname originalname newname

originalname
The present name of the remote file.

newname
The new name of the remote file. If the file specified by newname already exists, the new file replaces it.

This example renames the file SPORTSCAR.BMP in directory IMAGES on optical volume PICTURES to CAR.BMP:
REN /QOPT/PICTURES/IMAGES/SPORTSCAR.BMP
/QOPT/PICTURES/IMAGES/CAR.BMP

Note: On an iSeries 400 server, you cannot rename a file to a different file system.

Go To:
• FTP client subcommands [FTP client subcommands on page 37]
• FTP client syntax conventions [FTP client syntax conventions on page 66]

RESET (Reset)
FTP Client Subcommand

To clear the server reply queue, use the RESET FTP client subcommand:
REset

This subcommand resynchronizes the sequencing of the server subcommands and replies with the remote FTP server. Resynchronization may be necessary following a violation of the FTP protocol by the remote server.

Go To:
• FTP client subcommands [FTP client subcommands on page 37]
• FTP client syntax conventions [FTP client syntax conventions on page 66]
RMDIR (Remove Directory)
FTP Client Subcommand

RMDIR pathname

pathname
The name of a file directory, library, or other system-dependent file-group designator on the remote system. For hierarchical file system (HFS) directories, you can only delete empty directories. The server deletes Libraries unconditionally.

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

SENDPASV (Specify Whether to send a PASV Subcommand)
FTP Client Subcommand

To specify whether or not to send a PASV subcommand to the FTP server when doing a data transfer or issuing the DIR and LS subcommands, enter the SENDPASV FTP client subcommand:

SENDPASV [ 0 | 1 ]

If there is no parameter SENDPASV works like a toggle switch. The SENDPASV value toggles from 1 (ON) to 0 (OFF) or from 0 to 1.

When there is a parameter, the valid values are:

0 Do not send a PASV subcommand.
1 Send a PASV subcommand. This is the default.

iSeries 400 system default (on) is to send the PASV subcommand. When SENDPASV is off, then the server does not send the PASV subcommand.

Notes:
1. This subcommand supports RFC 1579, “Firewall-Friendly FTP.” Use of the PASV subcommand to establish a data connection is a better method when a data transfer must go through a firewall. In some scenarios, a data transfer through a firewall may not be possible without use of PASV.
2. Some FTP servers may not support the PASV subcommand. When this is the situation and SENDPASV is ON, then the FTP client will display a message that indicates that the server does not support PASV. The system will attempt to establish the data connection without sending the PASV subcommand.
3. When SENDPASV is OFF or disabled, then the server sends the PORT subcommand when SENDPORT is ON. See SENDPORT (Specify Whether to Sends a PORT Subcommand) "SENDPORT (Specify Whether to Sends a PORT Subcommand)" on page 60
4. FTP servers that do not support PASV are not compliant with RFC 1123.
Restriction
When connected to an FTP server through a SOCKS server, the SENDPASV subcommand may only be used before you issue any data transfer subcommand list directory subcommand. If you use SENDPASV after one of these subcommands, then the client will not be able to establish a data connection to the FTP server.

Once the client has issued a data transfer or list directory subcommand, then close the connection to the FTP server through a SOCKS server before you issues SENDPASV again.

You may use the SENDPASV subcommand when the FTP client is disconnected from an FTP server.

Go To:
- FTP client subcommands "FTP client subcommands" on page 37
- FTP client syntax conventions "FTP client syntax conventions" on page 66

SENDPORT (Specify Whether to Sends a PORT Subcommand)
FTP Client Subcommand

To specify whether or not to send a PORT subcommand to the FTP server when doing a data transfer or when issuing the DIR and LS subcommands. Enter the SENDPORT FTP client subcommand:

```
SENDPORT [ 0 | 1 ]
```

If there is no parameter SENDPORT works like a toggle switch. The SENDPORT value changes from 1 (ON) to 0 (OFF) or from 0 to 1.

When there is a parameter, the valid values are:

0       Do not send a PORT subcommand.
1       Send a PORT subcommand. This is the default.

Note:
1. Use SENDPORT only when you cannot establish a connection to the server without it. The indiscriminate use of SENDPORT may result in errors.
2. You may find it useful to not send the PORT subcommand to those systems that ignore PORT subcommands because they indicate that they have accepted the command.
3. The server does not send the PORT subcommand when the SENDPASV option is ON. See SENDPASV (Specify Whether to Send a PASV Subcommand) "SENDPASV (Specify Whether to Send a PASV Subcommand)" on page 59.

Go To:
- FTP client subcommands "FTP client subcommands" on page 37
- FTP client syntax conventions "FTP client syntax conventions" on page 66

SENDSITE (Specify Whether to Send a SITE Subcommand)
FTP Client Subcommand

To specify whether or not a SITE subcommand with record format information is automatically sent when doing a PUT or an MPUT operation, enter the SENDSITE FTP client subcommand in this format:
If there is no parameter, SENDSITE works like a toggle switch. The SENDSITE value changes from 0 (OFF) to 1 (ON) or from 1 to 0.

When there is a parameter, the valid values are:

0  Do not send a SITE subcommand. This is the default.
1  Send a SITE subcommand (containing record format information) prior to sending PUT and MPUT subcommands. Use this setting when transferring files to an IBM Virtual Machine server that uses the record format information that sends with the SITE subcommand.

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

SITE (Send Information Used by a Remote System)
FTP Client Subcommand

To send information that is used by the remote system to provide services specific to the remote system, use the SITE FTP client subcommand in this format:

SITE [parameters]

parameters  Dependent on the remote system.

To find the nature of these parameters and their syntax specifications, issue the HELP SERVER SITE subcommand. Some FTP servers do not support the SITE subcommand.

Note: The SITE subcommand is used by the PUT and MPUT subcommands to indicate the format and length of the records. By default, the PUT subcommand sends a SITE subcommand automatically. The NAMEFMT subcommand uses the SITE subcommand to indicate to the server whether names are in NAMEFMT 0 or NAMEFMT 1.

For more information, see SENDSITE (Specify Whether to Send a SITE Subcommand) "SENDSITE (Specify Whether to Send a SITE Subcommand)" on page 60.

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

STATUS (Retrieve Status Information from a Remote System)
FTP Client Subcommand

Status [name]

name  The name of the remote directory or file for which you request the status information. It is not a required parameter.
Note: The iSeries 400 FTP server application does not support this name parameter.

If there is no parameter, the server returns general status information about the FTP server process. This includes current values of all transfer parameters and the status of connections. The status information that is returned depends on the specific server implementation.

Go To:
- FTP client subcommands
- FTP client syntax conventions

STRUCT (Specify File Structure)
FTP Client Subcommand

To specify the structure of the data sent for a file, use the STRUCT FTP client subcommand in this format:

```
STRuct [F | R]
```

F  A file structure. The structure of a file is a continuous sequence of data bytes.
R  A record structure. The file transfers as a sequence of sequential records.

The structure of a file affects the transfer mode and the interpretation and storage of a file.

Go To:
- FTP client subcommands
- FTP client syntax conventions

SUNIQUE (Control Overwriting of Files)
FTP Client Subcommand

To control whether a file is overwritten when doing a PUT or MPUT subcommand, enter the SUNIQUE FTP client subcommand:

```
SUnique [ 0 | 1 ]
```

If there is no parameter, SUNIQUE acts like a toggle switch. The SUNIQUE value changes from 0 (OFF) to 1 (ON) or from 1 to 0.

When there is a parameter, the valid values are:

0  Overwrite the file if it exists. This is the default.
1  Create a new file with a unique name on the remote system instead of overwriting an existing file. The FTP server on the remote system sends the name of the created file back to the user.

Note: If the remote system is an iSeries 400 system, the server forms File.Mbr names by adding numbers to the end of the of the localfile that you specified in the PUT or MPUT subcommand. Thus, if the name NEWFILE.NEWMBR already exists on the remote system, the remote iSeries 400 system creates NEWFILE.NEWMBR1 and writes the data to it.
File names for other file systems, like HFS, work in a similar way. If the name already exists, a new file is created that consists of the specified file name and a number suffix. Thus, if the name `xfsname` already exists on the remote system, the remote iSeries 400 system creates `xfsname1`.

Go To:

- FTP client subcommands
- FTP client syntax conventions

**SYSCMD (Pass an iSeries 400 CL Command to Your Local iSeries 400 System)**

**FTP Client Subcommand**

To run a control language (CL) command on your local iSeries 400 system without leaving the FTP environment, use the SYSCMD FTP client subcommand in this format:

```
SYSCmd commandline
```

**commandline**

An iSeries 400 CL command. You can precede the command name with a `?` to get the prompt for the CL command. For example, if you enter:

```
SYSCMD ? SNDBRKMSG
```

you get the display for the Send Break Message (SNDBRKMSG) command.

If you want to see low level messages that result from your CL command, or if you want to enter multiple CL commands before returning to the FTP environment, use the iSeries 400 CALL QCMD command.

For example, to get to an iSeries 400 Command Entry display, enter this:

```
SYSCMD CALL QCMD
```

From the Command Entry display you can then call your application programs or enter CL commands. At the completion of your application program or the CL command, you return to the Command Entry display. From there you can display messages, start additional work on the system, or press F3 (Exit) or F12 (Cancel) to return to FTP.

You can enter iSeries 400 CL commands when you press F21 (CL command line) from the main FTP display. The server does not allow the F21 key when an exit program it is an addition to the FTP Client Request Validation exit point.

**Notes:**

1. Most server systems have a time-out period that ends the session if no activity occurs within a specific time period. If the command runs for longer than the time-out period, the server ends the connection with the client.
2. iSeries 400 supports the exclamation mark (!) as a synonym for the SYSCMD subcommand.
3. The SYSCMD subcommand passes to the iSeries 400 system as a CL command exactly what the user enters.

Go To:

- FTP client subcommands
- FTP client syntax conventions

**TYPE (Specify File Transfer Type)**

**FTP Client Subcommand**
To specify the file-transfer type, or the representation in which the transfer is to take place, use the TYPE FTP client subcommand in this format:

```
    B ccsid#
    C [ 1 ]
    I ]
```

A  Specifies the transfer type as the default (ASCII) transfer type. This has the same effect as the ASCII subcommand. The server does not associate any vertical format control with the file. It only supports the default format NON PRINT for ASCII. Use the ASCII transfer type or the transfer of text files, except when both systems use the EBCDIC type.

The default CCSID for TYPE A (ASCII) is the CCSID that is specified on the CCSID parameter of the STRTCPFTP command or FTP subcommand.

B  Shift JIS Kanji (CCSID 932)
B 1 Shift JIS Kanji (CCSID 932)
B 2 Extended UNIX Code Kanji (CCSID 5050)
B 3 JIS 1983 using ASCII shift-in escape sequence (CCSID 5054)
B 3 A JIS 1983 using ASCII shift-in escape sequence (CCSID 5054)
B 3 R JIS 1983 using JISROMAN shift-in escape sequence (CCSID 5052)
B 4 JIS 1978 using ASCII shift-in escape sequence (CCSID 5055)
B 4 A JIS 1978 using ASCII shift-in escape sequence (CCSID 5055)
B 4 R JIS 1978 using JISROMAN shift-in escape sequence (CCSID 5053)
B 5 Hangeul (CCSID 934)
B 6 Korean Standard Code KSC-5601, 1989 version (CCSID 949)
B 7 Traditional Chinese (5550) (CCSID 938)

C ccsid#

Specifies the transfer type to any CCSID (coded character set identifier) that is installed on the system. The CCSID number must follow C.

E  Specifies the transfer type as EBCDIC. This has the same effect as the EBCDIC subcommand. The server does not associate any vertical format control with the file. It only supports the default format NON PRINT for EBCDIC. Use the EBCDIC transfer type for the efficient transfer between systems that use EBCDIC as their internal character representation.

F  IBM EBCDIC Kanji (CCSID 5035)
F 1 IBM EBCDIC Kanji (CCSID 5035)
I  Specifies the transfer type as image. This has the same effect as the BINARY subcommand. With the image transfer type, data is a string of bits, packed into 8-bit bytes. The image transfer type is an efficient at storing and retrieving files and for transferring binary data such as object code. Data is transferred as is; there is no conversion.

If there are no parameters, the server displays the present setting for the TYPE subcommand.

Go To:
- FTP client subcommands [FTP client subcommands] on page 37
- FTP client syntax conventions [FTP client syntax conventions] on page 66
USER (Send Your User ID to the Remote System)
FTP Client Subcommand

User userid [password]

userid  Your logon name on the remote system.
password  Your password on the remote system. Specifying your password is optional. If you do not supply
your password when calling the USER subcommand, you receive a prompt to do so if the remote
system requires a logon password.

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

VERBOUSE (Control of Text Display of Error Reply Messages)
FTP Client Subcommand

To control the display of FTP server replies, use the VERBOUSE FTP client subcommand. The VERBOUSE
subcommand toggles the verbose switch on and off. When verbose is on, all server replies, including their
reply codes, are displayed. When set to off, certain server replies and reply codes are discarded and not
displayed.

Verbose

Go To:
• FTP client subcommands "FTP client subcommands" on page 37
• FTP client syntax conventions "FTP client syntax conventions" on page 66

FTP server reply status messages
When you enter subcommands during an FTP client session, status messages return to your display in a
3-digit code: xyz.

The first digit (x) tells you whether the response is good, bad, or incomplete. There are five values for the
first digit:
• 1yz = Good. The requested action is being initiated; another reply should follow.
• 2yz = Good. The requested action was successfully completed; a new request may be initiated.
• 3yz = Incomplete. The subcommand was accepted, but the requested action is being held pending
receipt of more information.
• 4yz = Incomplete. The server did not accept the subcommand. The requested action did not take place;
the error is temporary and you can request the action again.
• 5yz = Bad. The subcommand was not accepted, and the requested action did not take place.

The second digit (y) tells you the functional category of the response.
• x0z=Syntax. Refers to syntax errors, commands that aren’t appropriate for what you’re trying to do, and
unnecessary commands.
• x1z=Information. Refers to requests for information, such as status or help.
• x2z=Connections. Refers to the control or data connections.
• x3z=Authentication. Refers to the login process.
• x5z=File system. Refers to the status of the server in relation to the file transfer request.
The third digit (z) tells you a finer level of detail about the functional category.

Common reply codes and what they indicate are below. The message text may vary for different server systems.

<table>
<thead>
<tr>
<th>Code</th>
<th>What It Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>Restart the marker reply</td>
</tr>
<tr>
<td>120</td>
<td>Service is ready in nnn minutes</td>
</tr>
<tr>
<td>125</td>
<td>Data connection is already open; transfer is starting</td>
</tr>
<tr>
<td>150</td>
<td>File starting OK; about to open the data connection</td>
</tr>
<tr>
<td>200</td>
<td>Command OK</td>
</tr>
<tr>
<td>202</td>
<td>Command was not implemented; it is not used on this system</td>
</tr>
<tr>
<td>211</td>
<td>System status, or system help reply</td>
</tr>
<tr>
<td>212</td>
<td>Directory status</td>
</tr>
<tr>
<td>213</td>
<td>File status</td>
</tr>
<tr>
<td>214</td>
<td>Help message</td>
</tr>
<tr>
<td>220</td>
<td>Service is ready for a new user</td>
</tr>
<tr>
<td>226</td>
<td>Closing the data connection; the requested file action was successful</td>
</tr>
<tr>
<td>230</td>
<td>User is logged in</td>
</tr>
<tr>
<td>250</td>
<td>Requested file action was okay; action is completed</td>
</tr>
<tr>
<td>257</td>
<td>Path name was created</td>
</tr>
<tr>
<td>331</td>
<td>Password is required</td>
</tr>
<tr>
<td>332</td>
<td>Account is required</td>
</tr>
<tr>
<td>425</td>
<td>Cannot open the data connection</td>
</tr>
<tr>
<td>426</td>
<td>Connection is closed; the transfer ended abnormally</td>
</tr>
<tr>
<td>450</td>
<td>Requested file action was not taken; file busy</td>
</tr>
<tr>
<td>451</td>
<td>Requested action ended abnormally; local error in processing</td>
</tr>
<tr>
<td>452</td>
<td>Requested action was not taken; insufficient storage exists in system</td>
</tr>
<tr>
<td>500</td>
<td>Syntax error; command was unrecognized</td>
</tr>
<tr>
<td>501</td>
<td>Syntax error in the parameters or arguments</td>
</tr>
<tr>
<td>502</td>
<td>Command was not implemented</td>
</tr>
<tr>
<td>503</td>
<td>Bad sequence of commands</td>
</tr>
<tr>
<td>504</td>
<td>Command was not implemented for that parameter</td>
</tr>
<tr>
<td>530</td>
<td>Logon attempt was rejected</td>
</tr>
<tr>
<td>532</td>
<td>Need an account for storing files</td>
</tr>
<tr>
<td>550</td>
<td>Requested action was not taken; the file was not found (or no access)</td>
</tr>
<tr>
<td>551</td>
<td>Requested action ended abnormally; the page type is unknown</td>
</tr>
<tr>
<td>552</td>
<td>Requested file action ended abnormally; storage allocation was exceeded</td>
</tr>
<tr>
<td>553</td>
<td>Requested action was not taken; the file name is not allowed</td>
</tr>
</tbody>
</table>

**FTP client syntax conventions**

The FTP client subcommands described in this topic make use of these syntax conventions:

**Uppercase Letters**

Letters printed in uppercase in the syntax definitions for client subcommands are the minimum number of letters that you must enter. You can enter FTP client subcommands in either uppercase or lowercase.

**Lowercase Words or Hyphenated Terms**

Lowercase words or hyphenated terms, like remotefile and account-information, represent variables that you must substitute specific information.

**Brackets**

You can consider words, symbols, or phrases placed within brackets to be optional.

**Left Parentheses ( and Asterisks * **

You must enter left parentheses and asterisks exactly as they appear in the syntax definitions.
Braces {á}
Braces indicate a group of parameters, values, or variables that you may repeat.

Ellipsis . . .
Ellipses indicate that you can include zero or more repetitions of the preceding variable enclosed within brackets.

Vertical Bar |
A vertical bar between parameters or values indicates that you can specify one or the other, but not both, at one time. The vertical bars are within sets of brackets or braces.

More details on syntax:
- Enclosing subcommand parameters: Link to this information on how to use either an apostrophe (') or quotation marks ("" to enclose parameters.
- Default file names for client transfer subcommands: Link to this information for details on the default values.
- Naming files for transfer: Link to this information for details on the Localfile and Remotefile parameters.

Enclosing subcommand parameters
You can use either an apostrophe (') or quotation marks ("" to enclose subcommand parameters. To enclose an apostrophe within a parameter, you must enter it either as two consecutive apostrophes ("'") in a parameter that is enclosed by apostrophes. You must enter it as a single apostrophe in a parameter that is enclosed by quotation marks ("").

Similarly, if a quotation mark ("" is to be contained within a parameter, you must enter it in one of these ways:
- A single quotation mark (") in a parameter that is enclosed by apostrophes
- As two consecutive quotation marks (""") in a parameter that is enclosed by quotation marks.

You can use the apostrophe or quotation marks as follows:
1. If the apostrophe or quotation marks within the parameter are the same as the starting and ending delimiter, you must repeat the mark within the parameter. For example:
   
   "'ABCD'12345"  
   results in  ABCD'12345
   "ABCD""12345"  
   results in  ABCD"12345

2. If the starting and ending marks are not the same as the mark within the parameter, you do not repeat the mark. For example:
   
   "ABCD'12345"  
   results in  ABCD'12345
   'ABCD"12345"  
   results in  ABCD"12345

3. If both the apostrophe and quotation marks are within the parameter, you must choose one mark symbol as the delimiter. For example:
   
   "ABC'12""345" or 'ABC'12'345'  
   results in  ABC'12"345

More details on syntax:
- FTP client syntax conventions: Link to this information on FTP client syntax conventions.

Default file names for client transfer subcommands
The FTP client provides a default file name if the target file name for the PUT (Copy a File Member from the Local System to a File on a Remote System) on page 55, APPEND, and GET subcommands is omitted. Since you can specify source file names for the MPUT (Send Multiple File Members from the Local System to a Remote System) on page 53 and MGET (Copy Multiple Files from a Remote
The FTP server also generates target filenames for MPUT and MGET. See the Data Transfer Subcommands table below for the syntax of these subcommands. The table column labeled **Target** is the parameter for which a default name is provided.

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Source</th>
<th>Target</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPEND</td>
<td>local filename</td>
<td>[server filename]</td>
<td></td>
</tr>
<tr>
<td>PUT</td>
<td>local filename</td>
<td>[server filename]</td>
<td></td>
</tr>
<tr>
<td>GET</td>
<td>server filename</td>
<td>[local file name]</td>
<td>[Replace]</td>
</tr>
<tr>
<td>MPUT</td>
<td>local filename</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MGET</td>
<td>server filename</td>
<td></td>
<td>[Replace]</td>
</tr>
</tbody>
</table>

**PUT and APPEND**

For the PUT and APPEND subcommands, the rules for forming default names are divided into two categories:

- **iSeries 400 server case**
  - If the target file system is a library file system or a document library system, the default name complies with the naming rules for these systems, including their name format.
  - If the target file system is neither a library file system nor a document file system it is one of two names:
    - the default name is the name after the last slash in the source file name
    - the *same* as the source file name if there is no slash.

- **Non-iSeries 400 server case**
  - If the source file is a library file system file, then the default name consists of the *file name.member name*. If there is no member name, the file name is the default name.
  - If the source file is a document library services file, the default name is the file name and the extension.
  - If the source file is neither a library file system nor a document library services file, the name after the last slash in the source name is the default name. If there is no slash, the default name is the same as the source name.

If the server is an iSeries 400, then the server generates the default name in these subcommands using the same rules as applied for the PUT subcommand.

**GET and MGET**
If the server is not an iSeries 400, it bases the default name for the GET and MGET subcommands on the part of the source name that follows the last slash. If there is no slash, the entire source name is the default name. Here are the rules for forming default names:

• If the client file system is the library file system (iSeries 400 database), these rules apply:
  – If the remote file name contains a period (.), the characters preceding the period are truncated to 10 characters to form the local file name. The characters after the period are truncated to 10 characters to form the member name.
  – If the remote file name does not contain a period, both file and member names are set to the remote file name truncated to 10 characters to form the local file name.
  – If the name format is 1, the server adds the appropriate extensions to the file and member parts of the name.

• If the client file system is document library services, these rules apply:
  – If the remote name contains a period, the characters preceding the period are truncated to 8 characters. The characters after the period are truncated to 3 characters.
  – If the remote name does not contain a period, the name is truncated to 8 characters without an extension.

• For other file systems, the name after the last slash in the remote name is the default name.

Notes:
1. Save files do not have members, so default names for save files do not have a member part.
2. The server displays the default names when the DEBUG mode is on.

More details on syntax:
• FTP client syntax conventions “FTP client syntax conventions” on page 66

Naming files for transfer
The FTP client subcommands that you use for transferring data can have a localfile or a remotefile parameter or both. You can use these parameters to name the data you want to transfer. The transfer subcommands are:

APPEND APPEND (Append a Local File Member to a Remote File)” on page 41
localfile [remotefile]

DELETE DELETE (Delete a File on a Remote System)” on page 45
remotefile

GET GET (Copy a File from a Remote System to the Local System)” on page 46
remotefile [localfile]

MDELETE MDELETE (Delete Multiple Files on a Remote System)” on page 51
remotefiles

MGET MGET (Copy Multiple Files from a Remote System to the Local System)” on page 51
remotefiles

MPUT MPUT (Send Multiple File Members from the Local System to a Remote System)” on page 53
localfiles

PUT PUT (Copy a File Member from the Local System to a File on a Remote System)” on page 55
localfile [remotefile]
The names for the localfile and remotefile parameters can be either partially qualified or fully qualified. A partially-qualified name includes the name of the data itself as well as one or more names in the hierarchical sequence above the data. A fully-qualified name includes all names in the hierarchical sequence above the data.

When the name is partially qualified, the current working directory identifies the file to be processed. You can set the working directory on the local client system with the LCD (Change Working Library or Directory on Local System)* on page 48 subcommand. You can set the working directory on the remote server system with the CD (Change Working Directory or Library)* on page 43 subcommand.

The format of the localfile name parameters must conform to iSeries 400 file naming rules. The remotefile names must adhere to the file naming rules of the remote system.

More details on syntax:

- Enclosing subcommand parameters: Enclosing subcommand parameters* on page 67. You can use either an apostrophe (') or quotation marks (") to enclose parameters.
- Default file names for client transfer subcommands: Default file names for client transfer subcommands* on page 67. Link to this information about default file names for client transfer subcommands.
- FTP client syntax conventions: FTP client syntax conventions* on page 68. FTP client subcommands make use of these syntax conventions.

Working with the FTP server

The File Transfer Protocol (FTP) server function allows users on other computers to send or receive copies of files to or from your iSeries 400 server across a TCP/IP network. In addition, the FTP server provides functions for renaming, adding, and deleting files.

- Starting and ending the FTP server
  - Learn how to start and stop a server session.
- FTP server subcommands
  - See syntax and description information on the FTP server subcommands that are used to instruct the FTP server to establish connections, navigate libraries and directories, create files, delete files, and transfer files.

If you are having problems, see Troubleshooting FTP for information on troubleshooting the FTP server and client.

Starting and stopping the FTP server

The FTP server can be started and stopped using Operations Navigator. For instructions on how to access FTP, see Accessing FTP through Operations Navigator.

To start the FTP server, complete the following steps:
1. Expand your iSeries 400 server -> Network -> Servers -> TCP/IP.
2. Right-click FTP, and select Start.

Alternatively, you can start the FTP server with the CL command STRTCPSVR SERVER(*FTP).

To stop the FTP server, complete the following steps:
1. Expand your iSeries 400 server -> Network -> Servers -> TCP/IP.
2. Right-click FTP, and select Stop.

Alternatively, you can start the FTP server with the CL command ENDTCPSVR SERVER(*FTP).
FTP server subcommands

This topic is a reference of FTP server subcommands. The FTP client communicates with the server using server subcommands. Because a user does not typically communicate with the FTP server, we have provided the server subcommands, descriptions of what they do, their syntax conventions, and FTP reply status messages for your reference.

You may also need to refer to:

- FTP server syntax conventions on page 88
- Subcommands unique to iSeries 400 FTP server
- FTP server reply status messages on page 65
- FTP client subcommands on page 37: Use these subcommands to establish a connection with a remote FTP server, navigate libraries and directories, create and delete files, and transfer files.
- Transferring and receiving files with FTP on page 29: Access instructions on how to transfer files between a remote computer and your iSeries 400.

iSeries 400 FTP server system supports these subcommands:

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>What It Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABOR</td>
<td>Cancels the Previous Subcommand</td>
</tr>
<tr>
<td>ADDM</td>
<td>Adds a Member to a Physical File</td>
</tr>
<tr>
<td>ADDV</td>
<td>Adds a Member to a Variable-Length Member to a Physical File</td>
</tr>
<tr>
<td>APPE</td>
<td>Appends Data to a Specified File</td>
</tr>
<tr>
<td>AUTH</td>
<td>Defines the authentication mechanism used for the current FTP session.</td>
</tr>
<tr>
<td>CDUP</td>
<td>Changes Directory to the Parent Directory</td>
</tr>
<tr>
<td>CRTL</td>
<td>Creates a Library</td>
</tr>
<tr>
<td>CRTP</td>
<td>Creates a Physical File</td>
</tr>
<tr>
<td>CRTS</td>
<td>Creates a Source Physical File</td>
</tr>
<tr>
<td>CWD</td>
<td>Changes the Working Directory or Library</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Starts or Ends a Server Trace</td>
</tr>
<tr>
<td>DELETE</td>
<td>Deletes a File, a Member, or a Document</td>
</tr>
<tr>
<td>DLTF</td>
<td>Deletes a File</td>
</tr>
<tr>
<td>DLTL</td>
<td>Deletes a Library</td>
</tr>
<tr>
<td>HELP</td>
<td>Gets Information about FTP Server Subcommands</td>
</tr>
<tr>
<td>LIST</td>
<td>Lists Files or Directory Entries</td>
</tr>
<tr>
<td>MKD</td>
<td>Makes a Directory</td>
</tr>
<tr>
<td>MODE</td>
<td>Specifies a Format for Data Transmission</td>
</tr>
<tr>
<td>NLST</td>
<td>Lists the Names of Files or Directories</td>
</tr>
</tbody>
</table>
Subcommand  | What It Does  
--- | ---  
NOOP | Checks if Server is Responding  
PASS | Sends a Password to the Server  
PASV | Tells the Server to Passively Open the Next Data Connection  
PBSZ | Defines the largest buffer protection buffer size to be used for application-level encoded data sent or received on the data connection.  
PORT | Identifies the Data Port on which the Client Will Listen for a Connection  
PROT | Defines the protection used for FTP data connections  
PWD | Displays the Current Working Directory  
QUIT | Logs Off the User; Closes the Connection  
RCMD | Sends a CL Command to an FTP Server  
REIN | Re-starts a Session on a Server  
RETR | Retrieves Data from a Server  
RMD | Removes a Directory  
RNFR | Specifies a File to be Renamed  
RNTO | Specifies a New File Name  
SITE | Sends Information for a Server to Use  
STAT | Gets Status Information from a Server  
STOR | Saves Data on a Server and Replaces an Existing File  
STOU | Saves Data on a Server But Does Not Replace an Existing File  
STRU | Specifies the Structure of a File  
SYST | Prints the Name of the OS on the Server  
TIME | Sets the Time-Out Value for the FTP Server  
TYPE | Specifies the File Transfer Type  
USER | Sends a User Logon ID to the Server  
XCUP | Changes to the Parent Directory  
XCWD | Changes to the Working Directory  
XMKD | Creates a Directory  
XPWD | Displays the Current Directory or Library  
XRMID | Removes a Directory  

ADDM (Add Physical File Member)  
FTP Server Subcommand
ADDM parameters

parameters
   The parameters for this subcommand are the same as for the ADDPFM CL command.

For example, to add member BANANA to physical file GEORGE in library RLKAYS on an iSeries 400 server system, enter this:
ADDM FILE(RLKAYS/GEORGE) MBR(BANANA)

Go To:
   • FTP server subcommands: "FTP server subcommands" on page 71
   • FTP server syntax conventions: "FTP server syntax conventions" on page 88

ADDV (Add Physical File Variable Length Member)
FTP Server Subcommand

ADDV parameters

parameters
   The parameters for this subcommand are the same as for the ADDPVLM CL command.

For example, to add member POLEBEAN to physical file GEORGE in library RLKAYS on an iSeries 400 server system, enter this:
ADDV FILE(RLKAYS/GEORGE) MBR(POLEBEAN)

Go To:
   • FTP server subcommands: "FTP server subcommands" on page 71
   • FTP server syntax conventions: "FTP server syntax conventions" on page 88

APPE (Append to Existing File)
FTP Server Subcommand

The APPE FTP server subcommand accepts the transferred data and stores it in a file on the server system. If the file specified exists, it appends the data to that file; otherwise, it creates the specified file.

APPE filename

filename
   The file that will receive your on the server system.

Go To:
   • FTP server subcommands: "FTP server subcommands" on page 71
   • FTP server syntax conventions: "FTP server syntax conventions" on page 88

AUTH (Authorization)
FTP Server Subcommand

The AUTH FTP server subcommand defines the authentication/security mechanism that is used for the current FTP session. The syntax of this subcommand is:
AUTH [ TLS-C | TLS-P | TLS | SSL ]
Parameter values:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLS-C</td>
<td>Utilize the transport layer security (TLS) protocol as the security mechanism. The security settings for the data connection use the RFC2228 defaults; i.e., there is no implicit protection of the data connection.</td>
</tr>
<tr>
<td>TLS-P</td>
<td>Utilize the TLS protocol as the security mechanism. Also, implicitly protect the data connection (which is equivalent to the command sequence AUTH TLC-C, PBSZ 0, PROT P)</td>
</tr>
<tr>
<td>TLS</td>
<td>Synonym for TLS-C.</td>
</tr>
<tr>
<td>SSL</td>
<td>Synonym for TLS-P.</td>
</tr>
</tbody>
</table>

Note:
The TLS protocol is compatible with the secure sockets layer (SSL) protocol.

Go To:
- FTP server subcommands
- FTP server syntax conventions

CRTL (Create Library)
FTP Server Subcommand

CRTL parameters

parameters
The parameters for this subcommand are the same as for the CRTLIB CL command.

For example, to create a library that is called TESTTCP on a server iSeries 400 system, enter this:

CRTL TESTTCP

Go To:
- FTP server subcommands
- FTP server syntax conventions

CRTP (Create Physical File)
FTP Server Subcommand

CRTP parameters

parameters
The parameters for this subcommand are the same as for the CRTPF CL command.

For example, to create a physical file that is called MYFILE with a record length of 80 and no restrictions on the number of members, enter this:

CRTP FILE(RLKAYS/MYFILE) RCDLEN(80) MAXMBRS(*NOMAX)

Go To:
- FTP server subcommands
- FTP server syntax conventions
CRTS (Create Source Physical File)
FTP Server Subcommand

CRTS parameters

parameters
The parameters for this subcommand are the same as for the CRTSRCPF CL command.

For example, to create a source physical file that is called GEORGE in library RLKAYS, enter this:

```
CRTS FILE(RLKAYS/GEORGE)
```

Go To:
- FTP server subcommands [FTP server subcommands” on page 71]
- FTP server syntax conventions [FTP server syntax conventions” on page 83]

CWD (Change working directory or library)
FTP Server Subcommand

To change the working directory, library, or file group, use the CWD FTP server subcommand.

```
CWD directory
```

Go To:
- FTP server subcommands [FTP server subcommands” on page 71]
- FTP server syntax conventions [FTP server syntax conventions” on page 83]

DEBUG (Turn on the FTP Server Trace)

Note: Use the FTP server trace only for reporting software problems to IBM. You may affect system performance by this function.

FTP Server Subcommand

```
DEBUG
```

If the FTP server trace is not active, the server starts a trace. The server continues to run a trace until it receives another DEBUG subcommand or a QUIT subcommand. When it ends the trace, there may be a significant delay while it formats the trace data.

Go To:
- FTP server subcommands [FTP server subcommands” on page 71]
- FTP server syntax conventions [FTP server syntax conventions” on page 83]

DELE (Delete file or document)
FTP Server Subcommand

To delete a file, a member, or a document, use the CWD FTP server subcommand.

```
DELE remotefile
```
DLTF (Delete File)
FTP Server Subcommand

parameters
The parameters for this subcommand are the same as for the DLTF CL command.

For example, to delete file MYFILE in library RLKAYS, enter this:
DLTF FILE(RLKAYS/MYFILE)

HELP (Getting Help from an iSeries 400 Remote Server)
To get information about the FTP server subcommands, use the HELP subcommand in this format:
HELP [subcommand]

subcommand
The name of the server subcommand you want information about. For example, HELP ADDM will provide help information about how to add a member to a physical file on an iSeries 400 server system.

To determine the syntax of the ADDV subcommand that is used by iSeries 400 server, use the server subcommand:
HELP ADDV
LIST (File List)
FTP Server Subcommand

To get a list of directory entries, library contents, or files in a file group, use the LIST FTP server subcommand:

LIST [directory | name]

It lists only those files that FTP can transfer

Go to:
- FTP server subcommands
- FTP server syntax conventions
- SITE (Send Information Used by a Server System)
- LIST information in UNIX-style format
- LIST Information in iSeries 400 format

MKD (Make directory)
FTP Server Subcommand

To create or make a directory, use the MKD FTP server subcommand.

MKD directoryname

Go To:
- FTP server subcommands
- FTP server syntax conventions

MODE (Set Transfer Mode)
FTP Server Subcommand

To specify how you want bits of data transmitted, specify the mode, or data format, by using the MODE FTP server subcommand:

MODE [B | S]

B Specifies block mode. In this mode, data is a series of data blocks, preceded by one or more header bytes.

S Specifies stream mode. In this mode, data a stream of bytes. You can use any representation type with stream mode. This transfer mode is more efficient because the server does not transfer any data block information.

Notes:
1. Stream mode is the default transfer mode the iSeries 400 system uses and is the preferred mode.
3. If there is no parameter, the server returns a reply that indicates the present setting for MODE.

Go To:
• FTP server subcommands[FTP server subcommands” on page 71]
• FTP server syntax conventions[FTP server syntax conventions” on page 88]

NLST (Name List)
FTP Server Subcommand

To get a list of only the names of multiple files, a file group, a directory, or a library, use the NLST FTP server subcommand:

NLST [directory | name]

It lists only those files that FTP can transfer.

Go To:
• FTP server subcommands[FTP server subcommands” on page 71]
• FTP server syntax conventions[FTP server syntax conventions” on page 88]

NOOP (Obtain Server Response)
FTP Server Subcommand

The NOOP FTP server subcommand sends an “OK” reply to the client. It does not affect server processing in any other way. The client uses this command to determine if the server is connected and responding. Use the NOOP subcommand:

NOOP

Go To:
• FTP server subcommands[FTP server subcommands” on page 71]
• FTP server syntax conventions[FTP server syntax conventions” on page 88]

PASS (Password)
FTP Server Subcommand

PASS password

password
A string that specifies your password for the server system.

Note: The USER server subcommand must immediately precede the server subcommand PASS immediately.

Go To:
• FTP server subcommands[FTP server subcommands” on page 71]
• FTP server syntax conventions[FTP server syntax conventions” on page 88]

PASV (Use Passive Data Connection)
FTP Server Subcommand
To instruct this server to passively open the next data connection, use the PASV FTP server subcommand in this format:

PASV

Go To:
• FTP server subcommands
• FTP server syntax conventions

PBSZ (Protection Buffer Size)
FTP Server Subcommand

The PBSZ subcommand defines largest buffer size to be used for application-level encoded data sent or received on the data connection. The syntax of this subcommand is:

PBSZ value

where value is an ASCII character string representing a decimal integer.

Note:
RFC2228 requires that the PBSZ subcommand be issued prior to the PROT subcommand. However, TLS/SSL handles blocking of data, so '0' is the only value accepted.

Go To:
• FTP server subcommands
• FTP server syntax conventions

PORT (Data Port)
FTP Server Subcommand

To identify the port on which the client will listen for a data connection, use the PORT FTP server subcommand in this format:

PORT h1,h2,h3,h4,p1,p2

hn Represents the system IP address and is a character string that is a decimal value between 0 and 255.

pn Represents the TCP port number and is a character string that is a decimal value between 0 and 255.

To convert the p1 and p2 values to a TCP port number, use this formula:

port = ( p1 * 256 ) + p2

For example, in this PORT subcommand:

PORT 9,180,128,180,4,8
the port number is 1032 and the IP address is 9.180.128.180.

Note: After the server closes the connection, it cannot connect to the same client IP address and port number until a two-minute time delay has occurred as specified in TCP/IP RFC 1122. The server can make a connection to the same client IP address on a different port number without this restriction.

Go To:
- FTP server subcommands on page 71
- FTP server syntax conventions on page 88

PROT (Data Channel Protection Level) FTP Server Subcommand

The PROT subcommand defines the protection used for FTP data connections (which are used to transmit directory listings and file data). The syntax of this subcommand is:

PROT [ C I P ]

Parameter values:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Clear. The data connection carries &quot;raw data&quot; of the file transfer with no security applied.</td>
</tr>
<tr>
<td>P</td>
<td>Private. The data connection will use TLS/SSL, which provides Integrity and Confidentiality protection.</td>
</tr>
</tbody>
</table>

Go To:
- FTP server subcommands on page 71
- FTP server syntax conventions on page 88

PWD (Display Working Directory or Library) FTP Server Subcommand

The server returns a reply to the client with the name of the current directory or library when the PWD FTP server subcommand:

PWD

Go To:
- FTP server subcommands on page 71
- FTP server syntax conventions on page 88

QUIT (End an FTP Server Session) FTP Server Subcommand

The FTP server subcommand QUIT logs off the client user and closes the control connection. If a file transfer is in progress, the connection remains open until the file transfer is complete, and then the server closes it.

QUIT
RCMD (Send a CL Command to an FTP Server System)
FTP Server Subcommand

Use the server subcommand RCMD to run iSeries 400 control language (CL) commands on the FTP server system. The length of the RCMD subcommand string is up to 1000 characters. Because no prompting is available for the RCMD subcommand, the RCMD subcommand string must include all necessary parameters to run the CL command.

If the CL command called through the RCMD subcommand runs successfully, a message is displayed that states that the subcommand was successful. If an error occurred, it displays a message that states there was an error. The message does not include what the error was unless the error occurred because a library, file, or member name was not valid.

This is an example of using RCMD to run a Delete File (DLTF) command:

```
QUOTE RCMD DLT FILE(mylib/myfile)
```

`mylib` is the name of the library from which the file is to be deleted. `myfile` is the name of the file to be deleted.

REIN (Reinitialize Session between Systems)
FTP Server Subcommand

```
REINITIALIZE
```

The REINITIALIZE subcommand:
1. Allows the completion of any transfer in progress
2. Ends the USER session and removes all input/output and account information
3. Resets all server parameters to the default settings
4. Leaves the control connection open

RETR (Retrieve file)
FTP Server Subcommand

To retrieve data from the server system, use the RETR FTP server subcommand.
RETR remotefile

Go To:
- FTP server subcommands
- FTP server syntax conventions

RMD (Remove directory)
FTP Server Subcommand

To remove a directory, use the RMD FTP server subcommand.

RETR remotefile

Go To:
- FTP server subcommands
- FTP server syntax conventions

RNFR (Rename From)
FTP Server Subcommand

The RNFR FTP server subcommand renames files. It must be immediately followed by a RNTO (Rename To) server subcommand.

RNFR filename

filename
The name of the file you want renamed.

Note: The iSeries 400 server cannot rename a file to a different file system.

Go To:
- FTP server subcommands
- FTP server syntax conventions

RNTO (Rename To)
FTP Server Subcommand

The RNTO FTP server subcommand specifies the new file name when renaming files on the server system. It must immediately follow an RNFR subcommand, which specified the file name.

RNTO filename

filename
The name to which the file you want renamed.

Note: The iSeries 400 server cannot rename a file to a different file system.
SITE (Send Information Used by a Server System)
FTP Server Subcommand

To send information that is used by the server system or to provide services specific to the server system, use the SITE FTP server subcommand in this format:

SITE [parameters]

iSeries 400 FTP server supports these parameters for the SITE subcommand:

LISTFMT 0
The server returns information for the LIST subcommand in LIST Information in iSeries 400 format, which was introduced in V3R1M0. The iSeries 400 client supports both the iSeries 400 format and the UNIX-style format

LISTFMT 1
The server returns information for the LIST subcommand in LIST information in UNIX-style format. The name of the file is the last item of each line returned. The iSeries 400 client supports both the iSeries 400 format and the UNIX-style format

LISTFMT
Return a message that indicates the current FTP server LISTFMT setting.

Note:
If you wish to change the LISTFMT default on the server, then use the LISTFMT option of the CHGFTPA command. You can also use Operations Navigator to set this FTP server property:
1. In Operations Navigator, expand your iSeries 400 server—>Network—>Servers—>TCP/IP.
2. Double-click on FTP. The Properties screen will appear.
3. Click on the Initial Formats tab.
4. Under the File List heading, enable iSeries 400 or UNIX as the LISTFMT default on the server.
5. Click OK to accept the changes.

NAMEFMT 0
Use the LIBRARY/FILE.MEMBER name format. This name format is only for library file system database files.

NAMEFMT 1
Use the path name format. This name format is for all file systems that are supported by FTP that includes the library file system. Name format 1 must be used to work with all iSeries 400 file systems other than the library file system.

NAMEFMT
Return a message that contains the current server file name format.

Note:
You can configure the iSeries 400 FTP server the default NAMEFMT setting with the NAMEFMT option of the CHGFTPA command.
CRTCCSID *CALC
   New database files created during ASCII file transfers use the related default EBCDIC CCSID of the ASCII file transfer CCSID.

CRTCCSID *USER
   New database files created during ASCII file transfers use the current job CCSID. If this CCSID is 65535, the default CCSID is determined by the language id specifies the current job.

CRTCCSID *SYSVAL
   New database files created during ASCII file transfers use the CCSID that was specified by the QCCSID system value.

CRTCCSID [CCSID-number]
   Specify the CCSID when creating database files on the client during ASCII file transfers. The server validates this value.

CRTCCSID
   Display a message that contains the current FTP client CRTCCSID setting.

NULLFLDS 0
   The server does not allow transfer of database files that contain NULL fields. This is the default.

NULLFLDS 1
   The server allows transfer of database files that contain NULL fields.

   **Note:** Transfer of files that contain NULL fields requires both the client and server to have this setting enabled. If the server transfers a file that contains NULL fields to a non-iSeries 400 system, or if the transfer type results in codepage conversion of the data, then results are unpredictable.

NULLFLDS
   Return a message that indicates the current FTP server NULLFLDS setting.

TRIM 0
   Set Trim option to OFF. The server sends trailing blanks of database records.

TRIM 1
   Set Trim option to ON. The server does not send trailing blanks of database records when transferring database files that use file structure and stream mode. This is the default.

TRIM 2
   The server does not send trailing blanks of database records for all transfers, including record structure and block mode.

TRIM
   Returns a message that indicates the current setting of the FTP server Trim option.

**Notes:**
1. Prior to the availability of this subcommand, trailing blanks of QSYS.LIB file system records were always removed before transferring the file to the server system.
2. TRIM settings do not apply to TYPE I (binary) file transfers. Blanks are never trimmed for TYPE I file transfers, regardless of the TRIM setting.

**Go To:**
- FTP server subcommands [FTP server subcommands] on page 71
- FTP server syntax conventions [FTP server syntax conventions] on page 88

**STOR (Store File)**
FTP Server Subcommand
To save data on the server system and overlay an existing file, use the STOR FTP server subcommand in this format:

```
STOR remotefile
```

**Go To:**
- FTP server subcommands [FTP server subcommands](#)
- FTP server syntax conventions [FTP server syntax conventions](#)

### STOU (Store Unique) FTP Server Subcommand

To save data on the server system and not overlay an existing file, use the STOU FTP server subcommand:

```
STOU remotefile
```

The server generates a unique file name. The name assigned to the file will appear in the reply that is sent back to the client.

**Go To:**
- FTP server subcommands [FTP server subcommands](#)
- FTP server syntax conventions [FTP server syntax conventions](#)

### STRU (Specify File Structure) FTP Server Subcommand

To specify the structure of a file as a continuous sequence of data bytes, use the STRU FTP server subcommand in this format:

```
STRU [F | R]
```

- **F** A file structure. The file structure is a continuous sequence of data bytes.
- **R** A record structure. The file is a sequence of sequential records.

**Notes:**
1. 
2. The file structure affects the transfer mode and the interpretation and storage of a file.
3. If there is no parameter, the server returns a reply that indicates the present specification for file structure.

**Go To:**
- FTP server subcommands [FTP server subcommands](#)
- FTP server syntax conventions [FTP server syntax conventions](#)

### SYST (Identify the Name of the Operating System) FTP Server Subcommand

To obtain the name of the operating system on the server system, use the SYST FTP server subcommand:
The returned information is system dependent.

iSeries 400 server includes the TCP/IP version.

Here is an example server reply:
OS/400 is the remote operating system. The TCP/IP version is "V4R4M0".

Go To:

- FTP server subcommands
- FTP server syntax conventions

TIME (Set Time-Out Values for FTP Server)
FTP Server Subcommand

After the FTP control connection is established between the FTP client and the FTP server, the FTP server controls the time-out for this connection. This is the inactivity time-out value.

There is also a time-out value for the data connection, known as the transfer time-out.

The format of the TIME FTP server subcommand is:

```
TIME inactivity [transfer]
```

**inactivity**

The number of seconds the server waits before ending the connection with the client. Inactivity time-out values can range from 1-9,999,999 seconds. The default inactivity time-out value is 300 seconds.

**transfer**

The file transfer time-out in seconds. This parameter is optional. If you do not specify this parameter, then the server does not change the current value. Transfer time-out values can range from 1-9,999,999 seconds. The default transfer time-out value is 420 seconds.

For example, to set the inactivity time-out value of the FTP server to 1000 seconds, and keep the current value of the transfer time-out, enter this:

```
QUOTE TIME 1000
```

The TIME subcommand is not a standard FTP subcommand. It is iSeries 400 FTP server specific.

Go To:

- FTP server subcommands
- FTP server syntax conventions

TYPE (Specify Representation Type)
FTP Server Subcommand

To specify the file-transfer type or the representation in which the transfer is to take place, use the TYPE FTP server subcommand in this format:
A Specifies the transfer type as the default (ASCII) transfer type. The server does not associate any vertical format control with the file. The server only supports the default format NON PRINT for ASCII. The ASCII transfer type is for the transfer of text files, except when both systems use the EBCDIC type.

Note: The CCSID for TYPE A is the CCSID value of the FTP server configuration attributes. You can change these attributes with the CHGFTPA command.

B Shift JIS Kanji (CCSID 932)
B 1 Shift JIS Kanji (CCSID 932)
B 2 Extended UNIX Code Kanji (CCSID 5050)
B 3 JIS 1983 using ASCII shift-in escape sequence (CCSID 5054)
B 3 A JIS 1983 using ASCII shift-in escape sequence (CCSID 5054)
B 3 R JIS 1983 using JISROMAN shift-in escape sequence (CCSID 5052)
B 4 JIS 1978 using ASCII shift-in escape sequence (CCSID 5055)
B 4 A JIS 1978 using ASCII shift-in escape sequence (CCSID 5055)
B 4 R JIS 1978 using JISROMAN shift-in escape sequence (CCSID 5053)
B 5 Hangeul (CCSID 934)
B 6 Korean Standard Code KSC-5601, 1989 version (CCSID 949)
B 7 Traditional Chinese (5550) (CCSID 938)

C Specifies the transfer type to any CCSID (coded character set identifier) that is installed on the system. The CCSID number must follow C.

E Specifies the transfer type as EBCDIC. The server does not associate any vertical format control with the file. The server supports only the default format NON PRINT for EBCDIC. The EBCDIC transfer type is for efficient transfer between systems that use EBCDIC for their internal character representation.

F IBM EBCDIC Kanji (CCSID 5035)
F 1 IBM EBCDIC Kanji (CCSID 5035)

I Specifies the transfer type as image. With the image transfer type, data is a string of bits, packed into 8-bit bytes. The image transfer type efficiently stores and retrieves files and transfers binary data such as object code.

Go To:
- FTP server subcommands "FTP server subcommands" on page 71
- FTP server syntax conventions "FTP server syntax conventions" on page 88

USER (Send a User Logon ID to the Server)
The user profile on an iSeries 400 server system.

Note: If the USER subcommand is successful and the iSeries 400 server system is configured for password security, then the server sends a reply to the client requesting a password. The client sends the password to the server with the server subcommand PASS. There is no password prompt when the server system is running at security level 10.

Go To:
- FTP server subcommands
- FTP server syntax conventions

FTP server syntax conventions
The FTP server subcommands described in this topic make use of these syntax conventions:

**Uppercase Letters**
You must enter letters in uppercase exactly as shown in the syntax definitions for subcommands.
You can enter these letters in either uppercase or lowercase.

**Lowercase Words or Hyphenated Terms**
Lowercase words or hyphenated terms, such as (remotefile and account-information,) represent variables for which you must substitute specific information.

**Brackets [ ]**
You can consider words, symbols, or phrases placed within brackets to be optional.

**Left Parentheses ( and Asterisks * **
You must enter left parentheses and asterisks exactly as shown in the syntax definitions.

**Braces { }**
Braces indicate a group of parameters, values, or variables that you can repeat.

**Ellipsis . . . **
Ellipses indicate that you can include zero or more repetitions of the preceding variable enclosed within brackets.

**Vertical Bar |**
A vertical bar between parameters or values indicates that you can specify one or the other, but not both, at one time. You will see the vertical bars placed within sets of brackets or braces.

**Subcommands unique to iSeries 400 FTP server**
iSeries 400 FTP server subcommands include a special set of commands that are really abbreviated names of equivalent, but longer, iSeries 400 CL commands. The names of these special server subcommands must be four characters to comply with the FTP architecture limits. When iSeries 400 server receives these subcommands, this is how it interprets them:

- ADDM = ADDPFM (Add Physical File Member)
- ADDV = ADDPVLM (Add Physical File Variable Length Member)
- CRTL = CRTLIB (Create Library)
- CRTP = CRTPF (Create Physical File)
- CRTS = CRTSRCPF (Create Source Physical File)
- DLTF = DLT (Delete File)
• DLTL = DLTLIB (Delete Library)

In addition to these specific subcommands, you can use the FTP server subcommand RCMD to send any CL command to the server.

**Troubleshooting FTP**

If you detect a problem when using FTP, use the following flow chart to identify the cause after using the flow chart for general TCP/IP problems. The cause lists that follow list steps to help you identify the cause of the problem.

---

**FTP Problem Analysis**

**Cause List A**

1. Is there is a long delay between connecting to the iSeries FTP server and receiving a prompt for a user id? If so, check the configuration of the domain name server on your iSeries 400. The FTP server
performs a DNS query as soon as a new connection is received. DNS problems may cause the server to hang for several minutes before a response is received.

2. Check to see if an exit program has been added to the FTP Server Logon Exit Point. Refer to the Logon exit point for TCP/IP application servers subtopic. If yes, then check if the logon that is unsuccessful is allowed by the exit program.

3. Check to see if the remote logon requires a password if a password was requested. Some systems request a password, but the connection can fail because it is not required.

4. Set up a password on the remote system if required. You may have to restart if you change the security information on the system.

5. Check your user ID and password by attempting to sign on to your remote system. If you are unable to do so, contact the system owner to verify that your user ID and password are correct.

Cause List B
1. Make sure binary mode is in effect if you are transferring binary files.
2. Check to be sure the mapping tables on both the client and server systems are compatible. You need only do this if you are using your own mapping tables.
3. Check to see that the correct CCSID has been specified for the transfer. If not, use the TYPE or LTYPE subcommand to set the correct CCSID value before the transfer is performed.
4. Create a file on the system that you are planning to store data into. Set the proper record length, number of members, and number of increments. Try the data transfer again and verify that it was successful.

5. Make sure that you are authorized to use the file and the file members.
6. Check to see if the transfer file contains packed decimal or zoned decimal data.
7. If you are transferring a Save file, verify that the appropriate method was used.

Cause List C
1. Check file size limits on the remote system.
2. Check to see if the FTP server timer ended. The iSeries 400 server time-out value can be set using the QUOTE TIME command.
3. Use the NETSTAT command to verify that the *LOOPBACK interface is active. Then re-create the problem doing FTP LOOPBACK (iSeries 400-to-iSeries 400 internally).
   - If the problem cannot be recreated, it is probably a remote system problem.
   - If you can re-create the problem, do the following:
     a. If the problem is an FTP server problem, then start the FTP server trace using the TRCTCPAPP command.
     b. Create the problem again.
     c. End the FTP connection. Refer to the Starting and stopping the FTP server.
     d. End the FTP server trace using the TRCTCPAPP command.
     e. Find a spooled file with the following characteristics:
        - The file name is QTMFFTRC
        - The username associated with the file is the name of the user who issued the TRCTCPAPP command.
        The trace is a spooled file in the default output queue of the system associated with the FTP server job.
     f. Send in that spooled file.
     g. If the problem was on the iSeries 400 FTP client, a trace can be obtained using the DEBUG 100 client subcommand.
     h. When running the FTP client interactively, use the F6 (Print) key to create a spool file that contains a history of the FTP client subcommands entered, and the associated FTP server
replies. When the FTP client is run in batch unattended mode, then this history of subcommands and server replies is written to the specified OUTPUT file. For more details, see “FTP as Batch Job”.

If you are unable to correct your problem, see the following topics for more information:

- Materials required for reporting FTP problems
- Tracing the FTP server
- Tracing the FTP clients
- Creating a copy of an FTP server job log

Materials required for reporting FTP problems

Any FTP problem reported to IBM should include the following:

- A communications trace from the time of the failure (Request TCP/IP data only) formatted twice: once for ASCII and once for EBCDIC.
- If the FTP client or server has logged software error data, submit the data.

Note: The system value QSFWERRLOG must be set to "LOG for software error logging to take place. If an error occurs while QSFWERRLOG is set to "NOLOG, change the value to "LOG, try to re-create the error, and submit the logged software error data. If logged software error data is submitted, there is no need to perform a trace of FTP.

- The QTCPIP and any FTP server or FTP client job logs.
- The FTP client and FTP server debug traces.
- For FTP client problems, a spool file containing the FTP client session (which may be obtained by hitting the print (F6) key in the FTP session).
- If data integrity is the problem, then the file, member, or library causing the problem should be sent in along with a copy of the description of the file, member, or library.

Tracing the FTP Server

The FTP server can be traced from any iSeries 400 or non-iSeries 400 server that runs TCP/IP. There are two ways to trace the FTP server. The FTP server DBUG subcommand traces within an FTP server session. The Trace TCP/IP Application (TRCTCPAPP) command allows system wide tracing of all the FTP servers.

Tracing the FTP server with the DBUG subcommand

```plaintext
File Transfer Protocol
Previous FTP subcommands and messages:
Connecting to host name xxxxxxxnnn.xxxxxxxx.xxx.xxx at address n.nnn.nn.nn using port 21.
220-QTCP at xxxxxxxnnn.nnnnnnnn.nnn.nnn.
220 Connection will close if idle more than 5 minutes.
215 OS/400 is the remote operating system. The TCP/IP version is "V4R4M0".
> 331 Enter password.
230 TEST logged on.
250 Now using naming format "0".
257 "QGPL" is current library.

Enter an FTP subcommand.
====> quote dbug
```
To trace the FTP server:

1. Type `QUOTE DBUG` to start the trace.

2. Perform the FTP operation that you want to trace.

3. Type `QUOTE DBUG` again to end the trace. The trace creates a spooled file called QTMFFTRC. The default output queue contains the spooled file. The user is always the name of the user who was logged on to the FTP server when the trace was ended.

4. Type `QUIT` to end the FTP session.

5. Enter the following command to find the output queue:

   ```
   DSPSYSVAL QPRTDEV
   ```

   For example, the following display appears:
The printer device is also the name of the default system output queue.

6. Record the name of the printer device. In this example, PRT01 is the printer device.

7. Press F12 (Cancel) to return to the display where you entered the DSPSYSVAL command.

8. Type the following command:
   
   WRKOUTQ OUTQ(printer-device)
   
   Replace printer-device with the printer device recorded in the previous display. PRT01 is the output queue in this example. For example, the following display appears:

```
| Work with Output Queue |
| Queue: PRT01 Library: QGPL Status: RLS |
| Type options, press Enter. |
| 1=Send 2=Change 3=Hold 4=Delete 5=Display 6=Release 7=Messages |
| 8=Attributes 9=Work with printing status |
| Opt File User User Data Sts Pages Copies Form Type Pty |
| _ QTCPPRT QTCP QTMSMTP HLD 46 1 +STD 5 |
| _ QTMFFTRC QSECOFR HLD 44 1 +STD 5 |
```

9. Press F18 (Bottom) to get to the bottom of the spooled file list if More... appears on the display.

10. Find the last file named QTMFFTRC with the same user as the user who was logged on the FTP server when the trace was created.

11. Press F11 (View 2) to view the date and time of the file you want to work with.

12. Verify that you are working with the most recent spooled file, QTMFFTRC.

Indicate in the problem report that the trace was tried and it failed. Send whatever trace information there is with the problem report.

**Tracing the FTP server with the Trace TCP/IP Application (TRCTCPAPP) command**

*system wide* tracing of all the FTP servers.

The TRCTCPAPP command is provided specifically for trained service and development personnel. *SERVICE special authority is required to use this command. Use TRCTCPAPP in situations that require the capturing of trace data for service and development use. This command allows experienced personnel to dynamically start and stop tracing for applications.

With the use of TRCTCPAPP, trace information can be captured for the FTP TCP/IP application:

- Internal trace information can be captured for the AS/400 FTP server. The information that can be captured for the FTP server may be filtered using remote IP address and port or AS/400 user profile.
  - Only one trace can be active at a time on the system.

Here are two examples of the use of the TRCTCPAPP command:

**Example 1:**
TRCTCPAPP APP(*FTP) SET(*ON)

This will start tracing for all FTP servers. Tracing for all other TCP applications is not affected.

Example 2:
TRCTCPAPP APP(*FTP) SET(*CHK)

This command is used to check the status of the tracing for the FTP server job(s). Assume that the last command entered was: >
TRCTCPAPP APP(*FTP) SET(*ON) USER(JOECOOL)

The format of the response to this command would be a set of messages that would look similar to the following:
TCP45B7 TRCTCPAPP APP(*FTP) SET(*ON) USER(JOECOOL)
MAXSTG(*DFT) TRCFULL(*WRAP)
TCP45B1 Tracing active for *FTP.
TCP45B2 Data capture begun for *FTP.
TCP45B3 Data buffer wrapped for *FTP.

Tracing the FTP client

To produce an FTP client trace or display the subcommands sent to the FTP server, use the DEBUG FTP client subcommand. The DEBUG subcommand toggles the debugging mode. If an optional debug-value is specified, it is used to set the debugging level. When debugging is on, each subcommand sent to the server is displayed and preceded by the string '>>>'. The debug-value must be set to 100 to produce an FTP client trace.

```
DEBUG [debug value]
```

d debug value
If the debug-value is 0, debugging is off. If the debug-value is a positive integer, debugging is on. If no value is specified, the debug-value is toggled from zero to one or from a positive integer to zero.

100 Initiate an FTP client trace. The client continues running the trace until DEBUG is turned off or until the FTP client is ended. (When the trace is ended, there may be a significant delay while the trace data is formatted.)

Note: The FTP client trace should only be used for reporting software problems to IBM. System performance may be adversely affected by this function.

CRTDTAARA DTAARA(QTEMP/QTMFTP100) TYPE(*LGL)AUT(*USE)
If the QTMFTPD100 dataarea exists, then it will set the debug value to 100 and start an FTP client trace. The purpose of this capability is to enable FTP client debug traces to be done in those situations when an FTP client trace cannot be started by issuing the DEBUG 100 subcommand.

Creating a copy of FTP server job log
A copy of the FTP server job log may be required to obtain additional information about errors that occur on the FTP server. Issuing the following subcommand from the FTP client causes the server to create a spooled file of its job log:

```
QUOTE RCMD DSPJOBLOG
```

To obtain a copy of error messages written to the server job log, this subcommand must be issued after the error has occurred. The user may then inspect the job log using the WRKSPLF command.

This technique is recommended in those cases where the reply message returned to the client from the server only provides minimal information about an error occurring on the server machine. For example, this method is useful for obtaining details about I/O errors that occur on the server machine.

If the error prevents the FTP server job log from being obtained by the method described here, enter the following command to force a spooled job log to be created for each FTP session:

```
CHGJOBD JOBD(QTCP/QTMFTPS) LOG(4 00 *SECLVL)
```

Then recreate the scenario which causes the error. To restore the original job log behavior after obtaining the required data, enter the following command:

```
CHGJOBD JOBD(QTCP/QTMFTPS) LOG(4 00 *NOLOG)
```