GENTRAN:Server® for UNIX®

FTP Daemon User's Guide

Version 6.0

Sterling Commerce
An IBM Company

9939-6000-USER05-0000

Copyright Notice

February 2002

Trade Secret Notice

Sterling Commerce, Inc.
4600 Lakehurst Court Dublin, OH 43016-2000 * 614/793-7000

Table of Contents

Operating Instructions

Command Reference Guide 1

About This Guide

Contents

Welcome

Sterling Commerce's FTP Daemon is a File Transfer Protocol server designed to work with GENTRAN:Server® for UNIX®.

You can think of the GENTRAN:Server FTP Daemon as a tool that provides your Trading Partners direct access to their mailboxes while enabling you to limit their access to other components of your EDI system.

The FTP Daemon's facilities are easy to use. You do not need programming knowledge to accomplish most tasks. However, to fully understand and utilize the FTP Daemon's capabilities, you should be familiar with the UNIX operating system.

About this document

This document will introduce you to FTP Daemon concepts, suggest ways to configure the FTP Daemon, and show you how to install the FTP Daemon. It contains the following parts:

Chapter	Contents
1 and 2	Introduction to FTP Daemon concepts and features.
3	Pre-installation requirements and step-by-step instructions for installing and configuring the FTP Daemon. See Chapter 2 to learn about the steps required to implement the FTP Daemon.
4	A reference containing descriptions and examples of configuration and message files.
5	A reference containing Advanced Data Distribution mailbox commands used with the FTP Daemon.
6	A reference containing descriptions of the GENTRAN:Server FTP Daemon utilities used to monitor mailbox activity, shut down the FTP Daemon, and check the configuration files.
Appendix A	A mailbox commands reference that you can give to your Advanced Data Distribution users to help them use the FTP Daemon.
Index	An alphabetical list of subjects together with the page numbers where they appear in the text.

(Continued on next page)

Related publications

For information on GENTRAN:Server, refer to the *GENTRAN:Server Data Flow Administration Guide.*

For information about the UNIX operating system, see your UNIX manuals.

One source of information about FILE TRANSFER PROTOCOL (FTP), is *Request for Comments 959*. This document is available at http://www.w3.org/protocols/rfc959/.

Documentation conventions

The following documentation conventions are used in this guide:

italics	Italics are used to introduce new terms and to denote file names. Notes and asides also appear in italics.
bold	Boldface type is used to highlight important words and phrases. Menu selections, program names, field names, and FTP Daemon components often appear in bold type.

Conventions used in syntax statements

[ ]	Brackets indicate an optional part of a statement. Do not enter the brackets.
...	An ellipse indicates that the immediately preceding item can be repeated indefinitely. For example, **-e** ... means that you can repeat **-e** with other values.
( )	Parenthesis should be entered as shown. They are part of the syntax of a statement and are not special symbols.
<*abc*>	Substitute a value for any term that appears in lowercase italics surrounded by these symbols. For example, in email <*address*> you should replace *address* with a value.
under_score	An underscore bridges a multi-word term.

CHAPTER
1

What Is The GENTRAN:Server FTP Daemon?

Contents

Overview

Introduction

Sterling Commerce's FTP Daemon is a software tool that gives you more security while providing your Trading Partners and other users with direct access to their mailboxes.

Several FTP Daemon features enable you to control Advanced Data Distribution System users' access to your system. You can use the GENTRAN:Server FTP Daemon to do the following:

- Allow or refuse new connections
- Notify current users when a shutdown is planned
- Regulate the number of login attempts
- Limit the number of concurrent Advanced Data Distribution users
- Direct user access based on user ID or location.

Where the FTP Daemon resides

The FTP Daemon is installed as part of the GENTRAN:Server Advanced Data Distribution System. All FTP Daemon configuration files are located in *$EDI_mailbox/ftpd*. The FTP Daemon and all supporting programs are located in *$EDI_ROOT/bin*.

Reference
For additional information about where the Advanced Data Distribution System resides, see the Installation topic in the Getting Started chapter in the *GENTRAN:Server Advanced Data Distribution Guide*.

(Continued on next page)

What the FTP Daemon can do

The FTP Daemon is designed to give your customers more choices for direct accessing of their mailboxes, while its configurability lets you limit their penetration of your system.

This table describes what you can do with the FTP Daemon.

Feature	Description
Display customized messages	You can configure the Welcome Message file to display a unique welcome message when users log in. You can configure the FTP Daemon to display different messages for different users.
Log mailbox activity	Using this feature enables you to generate a series of reports based on the record of Advanced Data Distribution mailbox activity.
Allow on-the-fly compression	You can configure the FTP Daemon to allow Advanced Data Distribution users to receive files in compressed format.
Notify Advanced Data Distribution users if their password will expire soon	The FTP Daemon warns users when they need to change their passwords in order to assure continued access to their accounts. (Continued on next page)

(Contd) Feature	Description
Extend the Advanced Data Distribution command set	The use of some standard FTP Daemon commands jeopardizes Advanced Data Distribution System security, therefore, these commands have been disabled. However, should you choose to extend the command set, you can do so by listing the commands after the parameter *mbx_allowcmd* in the *ftpaccess* file. **Reference** For a list of FTP commands, see *Request for Comments 959*. This document is available at http://www.w3.org/protocols/rfc959/. **WARNING** **We strongly discourage the use of these commands. Sterling Commerce does not support problems resulting from the use of these commands.**
Accept only predefined files from Advanced Data Distribution users	The FTP Daemon accepts only files with prefixes you have defined as acceptable.

The FTP Daemon Features

Status Reports

Using the FTP Daemon, Advanced Data Distribution users can generate status reports for files in the following categories: dropped off, received, and waiting.

Requeuing Capability

Several mailbox commands enable users to requeue files based on specific criteria such as Mailbag ID and date.

Operating Environment

The GENTRAN:Server FTP Daemon operates in the UNIX operating environment. If you have GENTRAN:Server and the Advanced Data Distribution System correctly installed, there are no additional system requirements for running the FTP Daemon.

Reference
For more information about GENTRAN:Server UNIX operations, see the Operating Environment topic in the Understanding the Basics chapter of your *GENTRAN:Server Data Flow Administration Guide*.

Working with the FTP Daemon

Introduction

The program **ftpd** is the GENTRAN:Server FTP Daemon; this is the program responsible for establishing Advanced Data Distribution mailbox connections and handling all mailbox requests. The FTP Daemon product is composed of configuration files and several supporting programs.

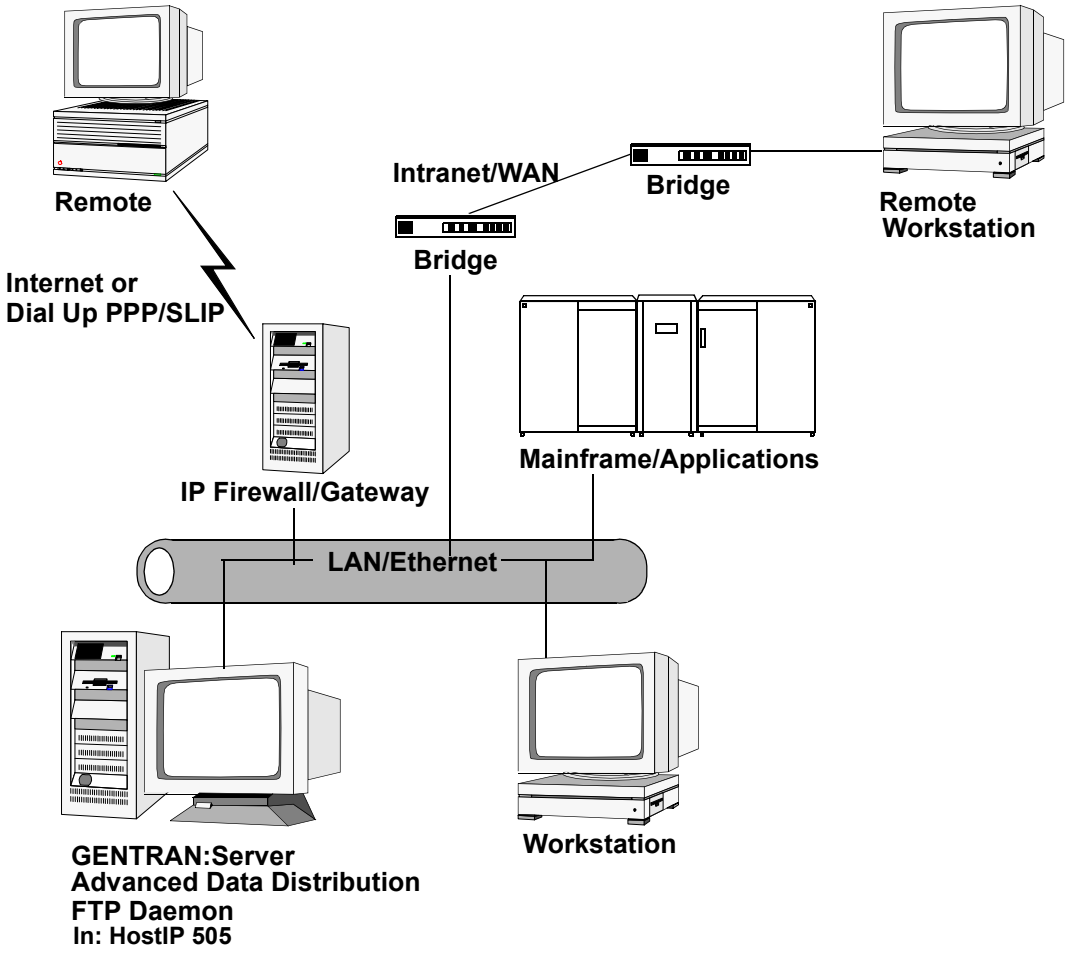
How the FTP Daemon facilitates user access

The Internet Services Daemon, **inetd** (also called the “super-server”), handles the FTP Daemon so that it is active only when someone needs it. **Inetd** is a daemon that listens on behalf of other daemons. It listens for a request on the port assigned to the GENTRAN:Server, and then spawns the FTP Daemon to service the request.

Once the FTP Daemon is activated and operating correctly, GENTRAN:Server Advanced Data Distribution System users can access their mailboxes. Advanced Data Distribution users must have access to their remote machine either through the Internet (directly or through a service provider) or through a wide or local area network, if your machine is directly reachable. To make the connection, Advanced Data Distribution users must start FTP client software and provide the name of your remote machine and the port number to which the FTP Daemon is assigned.

(Continued on next page)

Diagram The following diagram illustrates how the FTP Daemon works in an Internet/ Intranet environment.



(Continued on next page)

How the FTP Daemon can control user access

By configuring three files according to your organization's requirements, you can control the following:

- access to the Advanced Data Distribution System
- informational capabilities
- logging capabilities
- compression capabilities.

You also can control user access and capabilities on a per-user basis.

The files you must modify to configure the FTP Daemon correctly are the *ftphosts*, *ftpaccess*, and *ftpconversions* files.

Reference
See the Implementing the FTP Daemon chapter in this guide which contains instructions for configuring the FTP Daemon for an Advanced Data Distribution user. The files are discussed in detail in the Configuring the FTP Daemon Files chapter in this guide.

CHAPTER
2

Implementing the FTP Daemon

Contents

Overview

Introduction

The FTP Daemon, its supporting programs, and configuration files are installed as part of the GENTRAN:Server Advanced Data Distribution System, but you must complete some steps to activate the FTP Daemon. This chapter contains instructions for performing the required tasks to ensure proper operation of the FTP Daemon.

Before you complete the steps in this chapter, you will be directed to complete a Pre-Implementation Checklist to establish system information required for implementing the FTP Daemon.

Reference
For more information about the FTP Daemon configuration files, see the Configuring the FTP Daemon Files chapter. For detailed descriptions of FTP Daemon utilities, see FTP Daemon Utilities.

Setting up the FTP Daemon

During the implementation process, you will be asked to do several things to set up your FTP Daemon operations:

- Create a pseudo UNIX Advanced Data Distribution user account.

  You will use this account to define the userid, groupid, and permissions that will be assigned to files created or dropped off by Advanced Data Distribution users.
- Configure the FTP Daemon control file to process incoming files.

  You must modify the *ftpaccess* file. *Cfg_ftpaccess.sh* is a standard shell script that you can use to simplify your configuration of the *ftpaccess* file. This script resides in the directory *$EDI_mailbox/ftpd*. The script will prompt you for the minimum amount of information necessary to configure the *ftpaccess* file. The script uses the information you enter to create a new *ftpaccess* file. The original *ftpaccess* file is saved as *ftpaccess.old*.
- Modify system files */etc/inetd.conf* and */etc/services*.

  You must modify these files in order to facilitate the startup of the FTP Daemon.
- Configure GENTRAN:Server to process incoming files.

  You must create GENTRAN:Server queues and set up intelligent agents.

Complete the Pre-Implementation Checklist and the steps in the remainder of this chapter in the order in which they are presented.

Pre-Implementation Checklist

Complete this checklist and the accompanying chart before you implement your FTP Daemon installation.

From the GENTRAN:Server environment:		
1	*$EDI_ROOT* What is the full path?	
2	*$EDI_MAILDET* What is the full path?	
3	*$EDI_mailbox* What is the full path?	
4	What is the IP address of the machine where GENTRAN:Server resides? Look in */etc/hosts*	
5	What is the e-mail address of the person responsible for maintaining the FTP Daemon?	
6	What is the Port Number? (This is registered with the Internet Assigned Numbers Authority (IANA) as 505, but may be set to other values.) Look in */etc/services*	
7	What is the pseudo UNIX Advanced Data Distribution user Account Name? **Example:** *mbxusers*	
8	What service name (14-character maximum) do you want to be associated with the Port Number? (This service name is registered with the IANA as mailbox _lm, but may be set to other names, provided it is not already listed in the */etc/services* file.)	

File Prefix Reference Chart

Fill out this chart with all of the file prefixes the FTP Daemon will accept, along with related information you will need as you proceed with the implementation.

Note
A sample entry has been provided for you to follow.

File Prefix Reference Chart					
File Prefix	**Queue Name**	**Queue Resource Group**	**Queue Priority**	**IPC Flag**	**Intelligent Agent**
EDI850	edii	default	0	Y	iftp

The example above illustrates that the FTP Daemon will accept files with the prefix EDI850. A file with the EDI850 prefix will be added to the edii queue with a resource group of default and a priority of zero. The Intelligent Agent *in.ftp* is assigned to process the files in that queue and will be notified to start processing the file.

How to Set Up the FTP Daemon

Introduction

This topic describes the procedure for setting up the FTP Daemon for daily use.

Procedure

Use this procedure to configure the FTP Daemon for a user.

Step	Action
1	Log in to the machine where you installed the GENTRAN:Server Advanced Data Distribution System.
2	Add a new user to the */etc/passwd* file. You can use the program your system normally uses to add new users, or you can edit the */etc/passwd* file yourself. The user name to use is entered on Line 7 of the Pre-Implementation Checklist. **Important** You must have system administrator privilege to add a new user. **Example** */etc/passwd* entry for the Advanced Data Distribution account (with UID set to 215 and GID set to 20): mbxusers:*:215:20:FTPD mailbox Users:/dev/null: **Notes** You must make the group ID for your UNIX account the same as the GENTRAN:Server administrative group. If they are set to different GIDs, the permissions to access the Advanced Data Distribution files will be incorrect and Advanced Data Distribution mailbox commands will not be performed. In this example, the account is set with no password (so no one can log directly into the account) and the home directory of */dev/null* (the Advanced Data Distribution user actually is placed in his mailbox directory as opposed to */dev/null*). The startup program can be any valid shell as defined by the system administrator in the */etc/shells* file, or can be left empty (as in the above example) to use the default shell. **Reference** See the Understanding the Basics chapter of your *GENTRAN:Server Data Flow Administration Guide* for information about where GENTRAN:Server resides and the operating environment. (Continued on next page)

(Contd) Step	Action
3	Ensure that the account is enabled and that you know the password.
4	Create the queues listed in the File Prefix Reference Chart, using the Queue sub-menu selection from the GENTRAN:Server with Advanced Data Distribution Main Menu.
5	Ensure that the Intelligent Agent specified for the file prefix in the File Prefix Reference Chart is created and configured correctly. **Note** To process the files from the queue, the following three parameters listed in the Intelligent Agent Initialization file must be set to the values defined in the File Prefix Reference Chart: ◗ WORK_DIRECTORY must be set to the queue name listed in the File Prefix Reference Chart. ◗ WORK_TYPE must be set to 'q'. ◗ RESOURCE_GROUP must be set to the Queue Resource Group listed in the File Prefix Reference Chart. (Continued on next page)

(Contd) Step	Action
6	Enter the following commands to invoke the FTP access configuration script: cd $EDI_mailbox/ftpd sh cfg_ftpaccess.sh **System Response** The script prompts you for the following information: ◗ Paths to *$EDI_ROOT*, *$EDI_Advanced Data Distribution* and *$EDI_MAILDET*. The script can determine the paths to these directories providing the environment variables are set. If the environment variables are set, then the script asks you to confirm the directory locations; otherwise, you must enter the full path to these directories. ◗ The Advanced Data Distribution user's UNIX pseudo user ID that has been created (Line 7 on the Pre-Installation Checklist). ◗ The E-mail address of the person responsible for maintaining the FTP Daemon (Line 5 on the Pre-Installation Checklist). Users who encounter problems will be instructed to send E-mail to this address. ◗ The script will ask for the following information regarding processing of incoming files. (Refer to the File Prefix Reference Chart to locate the information.) ◗ Allowable file prefix (e.g., EDI850, TDC). This enables the FTP Daemon to accept only certain types of files. — The queue name that has been set up to process the incoming file. — The queue resource group that will be assigned to the incoming file. — The queue priority that will be assigned to the incoming file. The priority is a number between zero and nine, with zero as the highest priority and nine as the lowest priority. — Notification trigger. This will allow the intelligent agent to notify the downstream agent to start processing the file. After you have entered the information, the script will ask you if you would like to accept another file prefix.
7	Do you wish to accept another file prefix? ◗ If YES, enter Y and complete the rest of the information. ◗ If NO, enter N. (Continued on next page)

(Contd) Step	Action
8	Edit the */etc/inetd.conf* file to include the startup command for the FTP Daemon. **Important** You must have system administrator privilege to edit this file. The FTP Daemon is handled by **inetd**; it is active only when someone needs it. **Inetd** daemon listens on behalf of other daemons. It listens for a request on the port assigned to the GENTRAN:Server, and then spawns the FTP Daemon to service the request. **Format** <service name> stream tcp nowait root <path1>/bin/ftpd\ ftpd -m<path2> **Where** <*service name*> is the FTP Daemon service name (number 8 on the Pre-Implementation Checklist). <*path1*> is the path to *$EDI_ROOT* (Line 1 on the Pre-Installation Checklist). <*path2*> is the path to *$EDI_mailbox* (Line 3 on the Pre-Installation Checklist). **Note** You must specify the full path to the directory locations (environment variables are not allowed in the startup command). **Example** mailbox_lm stream tcp nowait root /usr/gentran/bin/ftpd\ ftpd -m/usr/gentran/mb In this example, mailbox_lm is the service name associated with the FTP Daemon. The path to *$EDI_ROOT* is */usr/gentran*. The path to *$EDI_mailbox* is */usr/gentran/mb*. (Continued on next page)

(Contd) Step	Action
9	Edit the */etc/services* file to map the FTP Daemon service name to the specified port number. The entry you place in the */etc/services* file associates the FTP Daemon's service name to a port number and the protocol it uses. **Inetd** watches the specified port for arriving data, which initiates FTP Daemon startup. **Important** You must have system administrator privilege to edit this file. **Example:** <service name> 505/tcp **Where** <*service name*> is the FTP Daemon service name (number 8 on the Pre-Implementation Checklist). The number 505 is the port number on which **inetd** listens. The number 505 has been assigned by the Internet Assigned Numbers Authority (IANA). The tcp indicates the protocol used by the FTP Daemon. **Example** mailbox_lm 505/tcp In this example, mailbox_lm is the service name associated with the port number 505. Tcp is the protocol used. (Continued on next page)

(Contd) Step	Action
10	Restart **inetd** by using the UNIX kill command to stop the inetd process. **Important** You must have system administrator privilege to restart **inetd**. **Note** Changes to the */etc/inetd.conf* and */etc/services* files do not take effect until you tell **inetd** to re initialize them. **Example** kill -1 16009 **Where** 16009 is the process ID of **inetd.** The -1 in the kill command causes **inetd** to reread its configuration file, *inetd.conf*. **Note** If your system administrator has configured *syslogd*, then after issuing the hangup, you must want to wait a minute and then check the log files for error messages related to your changes. (Continued on next page)

(Contd) Step	Action
11	Enter the following FTP command: ftp <IPAddress> <port#> **Where** <*IPAddress*> is the IP address of the machine where GENTRAN:Server Advanced Data Distribution resides (Line 4 on the Pre-Installation Checklist). <*port#*> is the port number (Line 6 on the Pre-Installation Checklist). **Example** ftp 123.45.6.78 505 **System Response** If your changes have taken effect, a message similar to the message shown below displays. Connected to 123.45.6.78 220 FTP server (Version wu-2.4(470) date time) ready. Name(username): **Note** If you do not see a message similar to the one above, re-examine the */etc/inetd.conf* and */etc/services* files to ensure that they are configured correctly, and restart your machine or reissue the kill command.
12	Log in with a Advanced Data Distribution user ID and password.
13	Test various FTP commands. **Reference** Descriptions of available commands are contained in the Operating Instructions chapter.
14	Log off the system.

Summary

You have now completed the process of setting up the FTP Daemon. Any user with a valid GENTRAN:Server Advanced Data Distribution mailbox ID can connect via the FTP Daemon.

CHAPTER
3

Configuring the FTP Daemon Files

Contents

Overview

Introduction

To operate correctly, the FTP Daemon requires proper configuration of several files. The following table describes the configuration files and explains their uses.

File	Use
ftphosts	To deny or allow individual machines access to the Advanced Data Distribution System.
ftpaccess	To configure access, informational, logging and miscellaneous capabilities to the Advanced Data Distribution System.
ftpconversions	To reflect correct paths to compression programs. (Compression conversions are stored in this file.)
welcome.msg	To greet users upon login.
limit.msg	To inform users that access is denied because the number of users for a given class has been reached.
deny.msg	To inform a user that access is denied.
mbxhelp.msg	To provide a list of valid Advanced Data Distribution mailbox commands.

The configuration files and message files described in the table reside in the directory *$EDI_mailbox/ftpd*. The correct parameters and paths must be specified in the *ftpaccess* file in order for the FTP Daemon to display the message files.

The configuration and message files are described in the following section, along with examples of each.

Configuration Files

Individual User Host Access File

Individual User Host Access File: ftphosts

The Individual User Host Access file enables you to allow or deny access to certain accounts in the Advanced Data Distribution System from various hosts.

Access is allowed or denied as described in the following table.

Situation	Access Denied	Access Allowed
The user is not listed in the file.		X
The file does not exist.		X
The user is listed in the file and the user's local address does not match <*addrglob*>.	X	
The user is listed in the file and the user's local address matches <*addrglob*>,	X (if <deny> is used)	X (if <allow> is used)

File Format

<allow|deny> <user> <addrglob> [<addrglob> ...]

Sample *ftphosts* File:

```
# Example host access file
#
# Everything after a '#' is treated as comment,
# empty lines are ignored

# Only allow host(s) matching <somehost.domain> to log in as <bartm>
   allow   bartm   somehost.domain
#
# Always deny host(s) matching <otherhost.domain>  or <131.211.32.*> to # log
in as <fred>
    deny    fred      otherhost.domain 131.211.32.*
#
# End example host access file
```

The FTP Daemon Configuration File

The FTP Daemon Configuration File: ftpaccess

The FTP Daemon Configuration file enables you to configure access, informational, logging, and miscellaneous capabilities to the Advanced Data Distribution System. You can configure the file with default values or with specific values for individual Advanced Data Distribution users.

The FTP Daemon configures itself when a user logs in by reading an *ftpaccess* file. The default location of this file is *$EDI_mailbox/ftpd/ftpaccess*. By putting an *ftpaccess* file in the *user* directory for any of your Advanced Data Distribution users, you can configure the FTP Daemon differently for individual Advanced Data Distribution users. When you put an *ftpaccess* file in a *user* directory, the FTP Daemon reads this file instead of the default *ftpaccess* file. Using this option enables you to do the following on a per-user basis:

- enable/disable features
- display different messages
- accept and route different files to Intelligent Agents.

Modifying the ftpaccess File

The *ftpaccess* file is configured with the minimum amount of information required to activate the FTP Daemon during implementation in Chapter 3 of this guide, when you used the *cfg_ftpaccess.sh* script. However, you can further customize the FTP Daemon by modifying the parameters stored in the *ftpaccess* file.

Sample ftpaccess file

```
# Example ftpaccess file
#
# Define <class> of users, with source addresses of the form <addrglob>.
# Multiple members of <class> may be defined. There may be multiple
# "class" commands listing additional members of the class. Failing to
# define a valid class for a host will cause access to be denied.
# <addrglob> may be a globbed
# domain name or a globbed numeric address.
# class <class> real <addrglob> [<addrglob> ...]


class   class1   real  192.42.0.*
class   class1   real  alpha
class   class2   real  ncr
class   class3   real  hpb
```

(Continued on next page)

```
#class   all      real  *
# Limit <class> to <n> users at times <times>, displaying <message_file>
# if user is denied access. Limit check is performed at login time only
# Failing to define a valid limit, or a limit of -1, is equivalent to
# unlimited.

# The limit command should only be listed in the default ftpaccess file.
# limit <class> <n> <times> <message_file>

limit   class1 2   Any   /usr/server/mb/ftpd/limit.msg
limit   class2 2   Any   /usr/server/mb/ftpd/limit.msg
limit   class3 3   Any   /usr/server/mb/ftpd/limit.msg

# Welcome message file will be displayed when user logs in
# message <message_file>

message      /usr/server/mb/ftpd/welcome.msg

# If the file pointed to <path> exists, the FTPD will check the
# file regularly to see if it is going to be shut down. If a shutdown is
# planned, new connections will be refused and user will be notified.
# The program ftpshut can be used to create the shutdown message file.
# shutdown <path>

shutdown  /usr/server/mb/ftpd/shutdown

# Defines the email address of the ftp archive maintainer. This
# string will be printed every time the %E token is used in message
# files.
# email <address>


email user@hostname

# Enables logging of individual commands by Advanced Data Distribution users.
The messages
# will be written to the system log maintained by the syslogd daemon.
# log commands real

log commands real

# Enables logging of file transfers. Logging of transfers TO the server
# (incoming) can be enabled separately from transfers FROM the server
# (outbound). <directions> is a comma-separated list of any of the two
# key words "inbound" and "outbound", and will respectively cause
# transfers to be logged for files sent to the server and sent from the
# server. Transfer information is stored in the file
# $EDI_mailbox/ftpd/xferlog
```

(Continued on next page)

```
# log transfers real <directions>
#xferlog is tied to system log and always uses host's time. CANNOT be
#changed with environment variable TZ.

log transfers real inbound,outbound

# Enables compress capabilities for any class matching any of
# <classglob>.
# The actual conversions are defined in the external file
# $EDI_mailbox/ftpd/ftpconversions.
# Only valid when retrieving files
# compress <yes|no> <classglob> [<classglob> ...]

compress        yes             class1

# Enables tar capabilities for any class matching any of <classglob>.
# The actual conversions are defined in the external file
# $EDI_mailbox/ftpd/ftpconversions.
# Only valid when retrieving files
# tar <yes|no> <classglob> [<classglob> ...]

tar             yes             class1

# Always deny access to host(s) matching <addrglob>.
# <message_file>will be displayed when a host has been denied access.
#<addrglob> may be "!nameserved" to
# deny access to sites without a working nameserver.
# deny <addrglob> <message_file>

deny alpha   /usr/server/mb/ftpd/deny.msg
deny hpb  /usr/server/mb/ftpd/deny2.msg



# Mailbox System help file will be displayed when a user logs in or
# issues a 'quote mbx_help' command.
# mbxhelp <path>

mbxhelp /usr/server/mb/ftpd/mbxhelp.msg

# Notify user if password will expire within <number> of days specified
# passwd_warn_msg <number>

passwd_warn_msg                 50

# After <number> login failures, log a "repeated login failures"
# message and terminate the FTP connection. Default value is 3.
# loginfails  <number>
```

(Continued on next page)

```
loginfails          3


# Define the directory path to EDI_ROOT
# EDI_ROOT    <path>

EDI_ROOT        /usr/server

# Define the directory path to EDI_MAILDET
# EDI_MAILDET <path>

EDI_MAILDET     /usr/server/mb

# Define mailbox user's UNIX pseudo userid
# edi_mbxru   <UNIX userid>

edi_mbxru           mbxusers

# Define additional commands for mailbox user
# A command set for the Advanced Data Distribution System has already been
defined.
# Additional commands are not necessary and should be avoided. If
# however you decide to add commands then caution should be taken to
# maintain system security.
# mbx_allowcmd <command> [<command>...]

mbx_allowcmd

# Define additional non-standard or UNIX specific commands for mailbox
# user
# A command set for the Advanced Data Distribution System has already been
# defined.
# Additional commands are not necessary and should be avoided. If
# however you decide to add commands then caution should be taken to
# maintain system security.
# mbx_allowsite  <command> [<command>...]

mbx_allowsite


# Determine what kind of data a mailbox will accept and routing
# information to Intelligent Agents.
# Define valid file prefixes and mapping information to queues
# mbx_prefix <file-prefix> <queue_name> [resource_group] [priority]
# Default value for resource_group = default
# Default value for priority = 9
# Default value for ipc_flag
```

(Continued on next page)

```
mbx_qprefix     EDI9    ftpi     default  2        y
mbx_qprefix     edi8    ftpi     default  4        n
mbx_qprefix     UDF     ftpu     default  1        n
mbx_qprefix     EDII    ftpi     default  3        n

#
#
#
# End example ftpaccess file
```

Conversions Database File

Conversions Database File: ftpconversions

The FTP Daemon Conversions Database file contains descriptions for compression conversions.

File Format:

%s:%s:%s:%s:%s:%s:%s:%s
1 2 3 4 5 6 7 8

Description

Each line in the file provides a description for a single conversion. Fields are separated by colons (:). Once the file is created, the user can request that files be sent compressed.

Field	Description
1	Not used. Leave blank.
2	Not used. Leave blank.
3	Not used. Leave blank.
4	Tostfix to add to real file
5	External command to do conversion
6	Types; T_REG for regular files T_ASCII for ascii files
7	Options for logging which conversion used; O_COMPRESS - file was compressed O_TAR - file was tar'ed
8	Description of conversion

Each of these fields is optional. If you choose not to use any of them, you must fill in the default value of %s.

(Continued on next page)

Sample Conversion File:

```
# Example ftpconversions file
#
 :   : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
 :   : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP

 :   : :.tar:/bin/tar -c -f - %s:T_REG|O_TAR:TAR
 :   : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|O_COMPRESS|O_TAR:TAR+GZIP
#
# End example ftpconversions file
```

Sample Compression Syntax

```
ftp> get OUT850-1.00011V.000188.Z
#
# The file will be sent to the user compressed
```

In this example, adding the z to the end of the filename causes the FTP Daemon to use the first entry in the *ftpconversions* file to compress the file. The file is compressed, then transferred to the user.

Message Files

Introduction

Standard message files are located with the configuration files in the *$EDI_mailbox/ftpd* directory. The messages are descriptive, but you can change the message content if you want to add more detail or otherwise alter them. You must specify the correct parameters and paths in the *ftpaccess* file for the FTP Daemon to display these files.

You can use the default values, or you can configure values specifically for individual users. To use a default message file, the message parameter in all of the *ftpaccess* files (default and *user*) must refer to the same message file.

To configure message files on a per-user basis, a user-specific *ftpaccess* file must be set up, with the message parameter referring to a different (user-specific) message file.

Note
While you may copy the message file into any directory you choose, we recommend you put it in the user directory. In any case, you must specify the correct path to the file in your user-specific ftpaccess file.

The *ftpaccess* file determines where message files reside.

Tokens can be used in the following message files:

- Welcome Message file
- Limit Message file
- Deny Message file

Tokens

The tokens are replaced automatically with a specific text string by the FTP Daemon. The valid tokens for the message files listed above are described in the following table, with descriptions of the substituted text strings.

Token	Description
%T	Local time (form Thr Nov 15 17:12:42 1998)
%C	Current working directory
%R	Remote host name
%L	Local host name

(Continued on next page)

(Contd) Token	Description
%U	Username given at login time
%M	Maximum allowed number of users in this class
%N	Current number of users in this class
%E	The FTP Daemon maintainer's E-mail address, as defined in the *ftpaccess* file

Welcome Message File

Welcome Message File: welcome.msg

The welcome message displays when a user logs in. You can customize the message according to your preference. The parameter *message* in the *ftpaccess* file determines where the *welcome.msg* file resides.

The tokens you can use in this file are listed in the table in the previous section of this chapter.

Due to variances in users' file transfer protocol, the message you specify may not display on all systems.

Sample *welcome.msg* File:

```
# Example welcome file
#
GENTRAN:Server Advanced Data Distribution FTP %T

Maximum allowed number of users in this class is = %M
Current number of users in this class = %N

Local  Host: %L
Remote Host: %R
User  Id: %U
Please send mail to %E if you experience any problems!
#
# End example welcome file
```

Limit Message File

Limit Message File: limit.msg

The limit message displays when a user is denied access because the number of users for a given class has been reached. The tokens described in the previous table can be used in the Limit Message file.

The parameter *limit* in the *ftpaccess* file determines where the *limit.msg* file resides.

Sample *limit.msg* File:

```
# Example limit file
#
Maximum allowed number of users in this class is = %M
Current number of users in this class = %N

Local     Time: %T
Local     Host: %L
Remote  Host: %R
User ID: %U
#
# End example limit file
```

Deny Message File

Deny Message File: deny.msg

The deny message displays when a host has been denied access. The tokens described in the previous table can be used in the Limit Message file.

The parameter *deny* in the *ftpaccess* file determines where the *deny.msg* file resides.

Sample *deny.msg* File:

```
# Example deny file
#
Machine denied access; %R
#
# End example deny file
```

Mailbox Help File

Mailbox Help File: mbxhelp.msg

The Advanced Data Distribution System help file displays when a user logs in or issues a 'quote mbx_help' command. Due to variances in users' file transfer protocol, the message you specify may not automatically display at login on all systems. In this case, issuing the quote command will display the file.

The parameter *mbxhelp* in the *ftpaccess* file determines where the *mbxhelp.msg* file resides.

Reference
For more information about the quote mbx_help command, see the Operating Instructions chapter.

Sample *mbxhelp.msg* File:

```
# Example mbxhelp file
# Listed below are the valid GENTRAN:Server

# Advanced Data Distribution System commands;
  Start session;
      user <user name>
  Send one file;
      put  <mailbox-file> [remote-file]
  Send multiple files;
      mput <mailbox-file*>
  Receive file;
      get  <mailbox-file> [local-file]
  Get multiple files;
      mget  <mailbox-file*>
  List mailbox files waiting to be picked up;
      ls
  Change mailbox password;
      quote mbx_cngpw         <oldpassed> <newpasswd>

  Requeue all previous received mailbox files;
      quote mbx_rq_all
  Requeue all previous received mailbox files received after date;
      quote mbx_rq_after      <date>
  Requeue all previous received mailbox files in the date range;
      quote mbx_rq_range      <low date> <high date>
  Requeue previous received mailbox files based upon mailbag ID;
      quote mbx_rq_mbag       <mbagid>
```

(Continued on next page)

```
Requeue previous received mailbox files based upon filename or file
prefix;
    quote mbx_rq_file       <filename | file prefix>
Generate a status report for all files dropped off;
    quote mbx_rpt_sa
Generate a status report for all files waiting to be or already picked
up;
    quote mbx_rpt_ra
Generate a status report for files waiting to be picked up;
    quote mbx_rpt_rw
Generate a status report for files already picked up;
    quote mbx_rpt_rp
Display valid filename prefixes:
    quote mbx_prefixes
Display this help file;
    quote mbx_help
End Session;
    quit
```

Notes:
Date is of the form [CC]YYMMDDHHMM or partial [CC]YYMMDDHHMM in the requeue commands.
The user can retrieve the status reports via get command.
#
End sample mbxhelp file

CHAPTER
4

FTP Daemon Utilities

Contents

Overview

Introduction

This chapter explains the GENTRAN:Server FTP Daemon utilities that enable the system administrator to monitor Advanced Data Distribution activity, shut down the FTP Daemon, and check to ensure that all configuration files exist.

Ftpcount

Description

The **ftpcount** program shows the current number of users and the maximum number of users for each class, which is defined in the default *ftpaccess* file.

Syntax

ftpcount -m $EDI_MAILBOX

Example

This example shows that one user is logged into class test, and the maximum number of users that can log into class test is two. Two users are logged into class prod with a maximum number of users equal to 4.

$EDI_ROOT/bin/ftpcount -m $EDI_MAILBOX
Service class test - 1 users (2 maximum)
Service class prod - 2 users (4 maximum)

Ftpshut

Description

Ftpshut closes down the FTP Daemon at a given time. The **ftpshut** program provides an automated shutdown procedure that enables a you to notify FTP Daemon users when the FTP Daemon is shutting down.

All of the shutdown information is stored in a *shutdown* file. The location of the *shutdown* file is specified with the parameter *shutdown* in the *ftpaccess* file. The FTP Daemon regularly checks for the existence of this file and will refuse new connections and notify users if a shutdown is planned. The FTP Daemon remains inactive until the *shutdown* file is removed.

Syntax

ftpshut -m $EDI_MAILBOX [-l min] [-d min] time [“warning-message ... “]

Where

time is the time at which the FTP Daemon will be shut down. You may use the word “now” to indicate an immediate shutdown, or specify a future time. Two formats can be used to specify the future time. Format 1 (+ number minutes) brings the FTP Daemon down in number minutes. Format 2 (HHMM) brings the FTP Daemon down at the time of day indicated, using a 24-hour clock format. You can shut down the FTP Daemon only between now and 23:59, if you use format 2.

New FTP Daemon access will be disabled 10 minutes before shutdown, or immediately if *time* is less than 10 minutes. This time may be adjusted using the -l flag.

All current connections will be disconnected five minutes before shutdown, or immediately if *time* is less than five minutes. This time may be adjusted using the -d flag.

The [`“warning-message ...”`] is formatted to be 75 characters wide. Tokens can be used in the warning message that will be replaced automatically with a specific text string by the FTP Daemon.

(Continued on next page)

Tokens

The valid tokens and a description of the substituted text string are described in this following table.

Token	Description
%s	Time system is going to shut down.
%r	Time new connections will be denied.
%d	Time current connections will be dropped.
%C	Current working directory.
%E	The FTP Daemon maintainer's E-mail address, as defined in the *ftpaccess* file.
%L	Local host name.
%M	Maximum allowed number of users in this class.
%N	Current number of users in this class.
%R	Remote host name.
%T	Local time (form Thu Nov 15 17:12:42 1990).
%U	Username given at login time.

Format

The format of the created *shutdown* file is:

<year> <month><day> <hour> <minute> <deny_offset> <disc_offset> <text>

Where
<*year*> is any year greater than 1970

<*month*> is 0-11

<*day*> is 1-31

<*hour*> is 0-23

<*minute*> is 0-59

<*deny_offset*> and <*disc_offset*> are the offsets in HHMM format before the shutdown time that new connections will be denied and existing connections will be disconnected.

(Continued on next page)

Example

```
# Example created shutdown file
#
1997 02 27 10 16 0003 0002
System is going to shut down  %s
New connections will be denied  %r
Current connections will be dropped  %d
#
# End example created shutdown file
```

Ckconfig

Description

The **ckconfig** program checks to ensure that the following configuration and log files exist:

- *ftpaccess*—the FTP Daemon configuration file
- *ftp.pids-%s*—this file is created automatically when users log in and is used to track the number of logged in users
- *ftpconversions*—the FTP Daemon conversions database file
- *xferlog*—this file is created automatically if you do transfer logging
- *ftphosts*—individual user host access file.

An OK message is displayed for files that exist, otherwise, an error message is displayed.

Syntax

```
ckconfig -m $EDI_MAILBOX
```

Example

```
Checking _PATH_FTPACCESS :: /usr2/srvr30/qa/mb/ftpd/ftpaccess
ok.

Checking _PATH_PIDNAMES :: /usr2/srvr30/qa/mb/ftpd/ftp.pids-%s
ok.

Checking _PATH_CVT :: /usr2/srvr30/qa/mb/ftpd/ftpconversions
ok.

Checking _PATH_XFERLOG :: /usr2/srvr30/qa/mb/ftpd/xferlog
ok.

Checking _PATH_FTPHOSTS :: /usr2/srvr30/qa/mb/ftpd/ftphosts
ok.
```

CHAPTER
5

Operating Instructions

Contents

Overview

Introduction

Advanced Data Distribution users can use several mailbox commands to control FTP Daemon operations and to record Advanced Data Distribution activity. This chapter contains detailed instructions for using Advanced Data Distribution commands to complete the following operations:

- Start FTP
- Log in
- Change the Advanced Data Distribution mailbox password
- Send files
- Receive files
- Requeue files
- List files
- Generate reports

The command descriptions in this chapter are organized according to the commands' functions, in the following categories: User Maintenance Commands, File Transfer and Requeuing Commands, and Report-Generating Commands.

User Maintenance Commands

Logging in to FTP—user

Starting FTP

To establish a connection between the host and the port number, enter the following FTP command:

ftp <IPAddress> <port#>

Where
<*IPAddress*> is the IP address of the machine where GENTRAN:Server Advanced Data Distribution resides (Line 4 on the Pre-Installation Checklist).

<*port#*> is the port number (Line 6 on the Pre-Installation Checklist).

System Response
The system displays a message similar to the following:

Connected to 123.45.6.78
220 FTP server (Version wu-2.4(470) date time) ready.
Name(username):

Log in and Authentication

When users log in to obtain access to the Advanced Data Distribution System, several checks process to determine whether to grant them access to their account. During the login process, the FTP Daemon authenticates Advanced Data Distribution users' access privileges according to the criteria outlined in the following list.

- You have set up a valid Advanced Data Distribution System mailbox.
- The user has entered a valid password.
- The password has not expired.
- The inactive period has not expired.
- Logins are enabled.
- Login attempts have not exceeded the *loginfails* value stored in the *ftpaccess* file.
- The Advanced Data Distribution machine is not denied access by the *ftphosts* file.

(Continued on next page)

- The number of users in the Advanced Data Distribution user's class has not exceeded the *limit* value in the *ftpaccess* file.
- A schedule shutdown has not been established as defined by the *shutdown* file created by **ftpshut**.
- The parameters *edi_mbxru* and *EDI_ROOT* are defined in the *ftpaccess* file.

An informative message displays if the login process fails.

Logging In

Users can log in to the FTP Daemon at any time. To access an account in a specific Advanced Data Distribution System, enter the **user** (Login User) command to log in to the FTP Daemon from any properly set up workstation.

Note
The **user** (Login User) command is valid only from an FTP Daemon prompt.

System Response
The system displays a prompt in which you must enter your corresponding GENTRAN:Server Advanced Data Distribution mailbox password.

Syntax

user <user name>

Log in process

The **user** command logs the Advanced Data Distribution user in and starts the mailbox session. This table describes the actions that occur during the log in process.

Stage	Description
1	If the user has an *ftpaccess* file in his or her *user* directory, the FTP Daemon reads this file after the default *ftpaccess* file.
2	The owner and group access of the FTP Daemon are determined by the pseudo Advanced Data Distribution UNIX account. The *edi_mbxru* parameter in the *ftpaccess* file specifies the pseudo UNIX login account. **Note** The UNIX account should have been created with a restricted access.
3	The user has access to display the files in *./togo* directory of the mailbox.

(Continued on next page)

(Contd) Stage	Description
4	A mailbag ID is generated for the current mailbox session.
5	The starting mailbox session message is recorded in the Mail Detail log.
6	The mailbox Help file containing valid mailbox commands is displayed.
7	A welcome message is displayed.
8	A warning message is displayed if the password will expire within the number of days defined by the *passwd_warn_mes* parameter in the *ftpaccess* file.

Note
Due to variances in users' File Transfer Protocol configurations, the mailbox Help file and welcome message may not display on all systems. In order to see the message and file, users should use the quote mbx_help command.

Reference
See the Display Mailbox Help File—mbx_help topic later in this chapter for instruction on accessing the file.

Change Password—mbx_cngpw

Description

The change password command enables users to change their passwords. All activity is recorded in the Mail Detail log. The maximum length of the new password is nine characters.

A message similar to “200 ACCEPT - PASSWORD CHANGED “ is displayed if the password has been changed successfully, otherwise, an error message is displayed. A password change can fail for any of the following reasons:

- the old password was incorrect
- an error occurred reading the mailbox file
- the length of the new password exceeds nine characters

Syntax

quote mbx_cngpw <oldpasswd> <newpasswd>

Example 1

Change password from test to test2

```
ftp>  quote mbx_cngpw test test2
200 ACCEPT - PASSWORD CHANGED.
```

Example 2:

Unable to change password because length of *newpasswd* exceeds nine characters

```
ftp> quote mbx_cngpw test2 longnewpasswd
200-    Length of newpwd can not exceed 9 characters
200 REJECT - PASSWORD CHANGE FAILED.
```

Example 3:

Unable to change password because *oldpasswd* is incorrect

```
ftp> quote mbx_cngpw junk new
200- passw, oldpasswd incorrect
200 REJECT - PASSWORD CHANGE FAILED.
```

(Continued on next page)

Example 4: Unable to change password due to an error reading the file *$EDI_mailbox/mbox.dat.*

```
ftp>  quote mbx_cngpw sadf asdf
200-    passw, abc_ifopen(mbox), iserrno = 2
200 REJECT - PASSWORD CHANGE FAILED.
```

Display Mailbox Help File—mbx_help

Description

This command displays the mailbox help file as defined in the *ftpaccess* file. Users whose File Transfer Protocol configuration impedes the automatic display of this file at login will use this command if they want to view the file.

Syntax

quote mbx_help

End Session—quit

Description

This command terminates the FTP session and exits. An end session record is placed in the Mail Detail log.

Syntax

quit

File Transfer and Requeuing Commands

Send One File—put

Description

The **put** command (Send One File) enables users to send a file to their mailbox.

Syntax

put <mailbox-file> [remote-file]

Using this syntax, you can send a particular mailbox file. If remote-file is unspecified, then the mailbox-file name is used as the remote-file name.

The put Process

This table describes the actions that occur when the **put** command is used.

Stage	Description
1	An error message is displayed if the user is trying to send a file outside of his or her mailbox directory.
2	The FTP Daemon determines whether to accept the user's file based on the file prefix. The FTP Daemon extracts the file prefix and compares it against all prefixes stored in the *ftpaccess* file. If a match is found, a copy of the file is sent from the FTP client to the Advanced Data Distribution System, otherwise the following message is displayed: fn: Permission denied. Not a valid file-prefix.
3	The file is renamed to <*filename*>.<*mailbagID*>.<*uniqueID*>.
4	The file is moved to the *toco* directory and registered as queued.
5	The file is routed to the Intelligent Agent's queue as defined by the mapping of file type to Intelligent Agent in the *ftpaccess* file.
6	The file is moved from the *toco* directory to the *came* directory and registered as dequeued.

(Continued on next page)

(Contd) Stage	Description
7	All activity is recorded in the Mail Detail log.
8	If the file was queued successfully to an Intelligent Agent, the following message is displayed: ********** * REGISTRATION TO Advanced Data Distribution System SUCCESSFUL * ********** otherwise, the following message is displayed: ********** * REGISTRATION TO Advanced Data Distribution System FAILED * **********

Example 1 Send file EDIDATA1

```
ftp>  put EDIDATA1
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDIDATA1.00011F.000171.
226-Transfer complete.

************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
************************************************************
226 Done.
236 bytes sent in 0.012 seconds (20 Kbytes/s)
```

(Continued on next page)

Example 2 Send file EDIDATA1 and store it as EDIDATA2 on remote host

```
ftp> put EDIDATA1 EDIDATA2
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDIDATA2.00011F.000172.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
236 bytes sent in 0.00098 seconds (2.4e+02 Kbytes/s)
```

Example 3 Unable to put files outside of their mailbox directory

```
ftp> put EDIDATA1 /etc/EDIDATA2
200 PORT command successful.
553-Permission denied. Unable to store file outside of Advanced Data Distribution
System;
/etc/EDIDATA2

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM FAILED *
**********************************************************************************************
553 Done.
```

Example 4 Advanced Data Distribution System not set up to receive data file

```
ftp> put junk
200 PORT command successful.
553-junk: Permission denied. Not a valid file-prefix.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM FAILED *
**********************************************************************************************
553 Done.
```

Send Multiple Files—mput

Description

The **mput** (Send Multiple Files) command enables users to send multiple files to their mailboxes. The operations of this command are identical to the operations of the **put** (Send One File) command. All activity is recorded in the Mail Detail log.

Syntax

mput <*mailbox-file**>

The commands in the following examples send all mailbox files of a particular type.

Example 1

Send all files that begin with EDI850

```
ftp> mput EDI850*
local: EDI850-1 remote: EDI850-1
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDI850-1.00011H.00017E.
226-Transfer complete

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
local: EDI850-2 remote: EDI850-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDI850-2.00011H.00017F.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
```

(Continued on next page)

Example 2 Send all files that begin with TDCC204

```
ftp> mput TDCC204*
local: TDCC204-1 remote: TDCC204-1
200 PORT command successful.
150 Opening BINARY mode data connection for
      TDCC204-1.00011H.00017G.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
local: TDCC204-2 remote: TDCC204-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      TDCC204-2.00011H.00017H.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
ftp>
```

Example 3 Send all mailbox files

```
mput *
```

Receive File—get

Description

The **get** (Receive File) command enables users to receive a file from their mailbox.

Syntax

get <mailbox-file> [local-file]

Using this syntax, users can retrieve a particular mailbox file. If local-file is unspecified, then the specified mailbox-file name is used as the local-file name.

This table describes the actions that occur when the **get** command is used.

Stage	Description
1	An error message is displayed if users try to retrieve files outside of their mailbox directory.
2	The FTP Daemon determines if the file is registered in the Advanced Data Distribution System. An error message displays if the file is not registered; otherwise, the FTP Daemon transfers a copy of the mailbox file to the user.
3	The mailbox file is moved from the user's *togo* directory to the *gone* directory and registered as dequeued.
4	All activity is recorded in the Mail Detail log.

Example 1

Retrieve file OUT856-1.00011V.000191.

```
ftp> get OUT856-1.00011V.000191
200 PORT command successful.
150 Opening BINARY mode data connection for
    OUT856-1.00011V.000191 (5 bytes).
226 Transfer complete.
5 bytes received in 0.00098 seconds (5 Kbytes/s)
```

(Continued on next page)

Example 2 Retrieve file OUT850-2.00011V.000189 and store it on the local host as /usr/users/data/OUT850-2.

```
ftp> get OUT850-2.00011V.000189 /usr/users/data/OUT850-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-2.00011V.000189 (5 bytes).
226 Transfer complete.
5 bytes received in 0.00098 seconds (5 Kbytes/s)
```

Example 3 Unable to retrieve file because it is not registered in the Advanced Data Distribution System.

```
ftp> get junk
200 PORT command successful.
550-    deque, isread, iserrno=111
550 DEQUEUE FAILED junk
```

Example 4 Unable to retrieve files outside of the user's mailbox.

```
ftp> get /usr/users/data/OUT850-2
200 PORT command successful.
550 Permission denied. Unable to retrieve file outside of
      Advanced Data Distribution System;
/usr/users/data/OUT850-2
```

Receive Multiple Files—mget

Description

The **mget** (Receive Multiple Files) command enables users to receive multiple files from their mailbox. The operations of this command are identical to the operations of the **get** command. All activity is recorded in the Mail Detail log.

Syntax

The following commands retrieve all mailbox files of a particular type.

mget <mailbox-file*>

Example 1

Retrieve all files that begin with OUT850.

```
ftp> mget OUT850*
local: OUT850-1.00011V.000188 remote:
        OUT850-1.00011V.000188
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-1.00011V.000188
(11 bytes).
226 Transfer complete.
11 bytes received in 0 seconds (0.011 Kbytes/s)
local: OUT850-2.00011V.000189 remote:
        OUT850-2.00011V.000189
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-2.00011V.000189
(5 bytes).
226 Transfer complete.
5 bytes received in 0.034 seconds (0.14 Kbytes/s)
```

(Continued on next page)

Example 2 Retrieve all files that begin with OUT856.

```
ftp> mget OUT856*
local: OUT856-1.00011V.000191 remote:
        OUT856-1.00011V.000191
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT856-1.00011V.000191
(5 bytes).
226 Transfer complete.
5 bytes received in 0 seconds (0.0049 Kbytes/s)
local: OUT856-2.00011V.000192 remote:
        OUT856-2.00011V.000192
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT856-2.00011V.000192
(5 bytes).
226 Transfer complete.
5 bytes received in 0 seconds (0.0049 Kbytes/s)
```

Example 3 Retrieve all mailbox files.

```
mget *
```

List Mailbox Files Waiting to be Picked Up—ls

Description

This command lists mailbox files waiting to be picked up. These files are located in the user's *togo* directory. Users are not allowed to view files outside of their mailbox directory. If users try to view a file not located in their mailbox directory, an error message displays.

Syntax

ls

Example 1

View all files waiting to be picked up.

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
OUT850-1.00011V.000188
OUT850-2.00011V.000189
OUT856-1.00011V.000191
OUT856-2.00011V.000192
rcv.rpt
226 Transfer complete.
```

Example 2

View (long format) all files waiting to be picked up.

Note
The filenames may continue onto the next lines differently on your screen display.

```
ftp> ls -l
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 40
-rw-rw-r--1 data users          11 Apr 8 19:28 OUT850-1.00011V.000188
-rw-rw-r--1 data users          5 Apr  8 18:59 OUT850-2.00011V.000189
-rw-rw-r--1 data users          5 Apr  8 19:00 OUT856-1.00011V.000191
-rw-rw-r--1 data users          5 Apr  8 19:00 OUT856-2.00011V.000192
-rw-rw-r--1 mbxusers users646 Apr  8 19:29 rcv.rpt
226 Transfer complete.
```

(Continued on next page)

Example 3 Users are not allowed to view files outside of their mailbox directory.

```
ftp> ls /etc
200 PORT command successful.
550 Permission denied. Unable to view files outside of Advanced Data
Distribution System; /etc
```

Requeue All Mailbox Files—mbx_rq_all

Description

This command requeues all previously received mailbox files. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_all

Example

In this example, the failed requeue occurred because the entry was marked.

```
ftp> quote mbx_rq_all
200-Requeue ALL operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Failed; can not requeue marked entry;
                EDIDATA.00000E.000001014
    Requeue Successful; TDCC204-1.0000DI.000001976
    Requeue Successful; TDCC204-2.0000DI.000001978

200 Requeue ALL operation completed.
ftp>
```

Reference

For more information about marked entries, see the *GENTRAN:Server Advanced Data Distribution User's Guide*.

Requeue Files Received After Date—mbx_rq_after

Description

This command requeues all previously received mailbox files that were received after the specified date. The date is in the following format, full or partial: [CC]YYMMDDHHMM. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_after <date>

Example

```
ftp> quote mbx_rq_after 960331
200-Requeue AFTER operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Failed; can not requeue marked entry;
                EDIDATA.00000E.000001014
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970

200 Requeue AFTER operation completed.
ftp>
```

Requeue Range—mbx_rq_range

Description

This command requeues all previously received mailbox files in the date range specified. The date is in the following format, full or partial: [CC]YYMMDDHHMM. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_range <low date> <high date>

Example

```
ftp> quote mbx_rq_range 9604051646 960407
200-Requeue RANGE operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970
    Requeue Successful; EDIDATA.0000DH.000001974

200 Requeue RANGE operation completed.
ftp>
```

Requeue Mailbag Id—mbx_rq_mbag

Description

This command requeues all previously received mailbox files based on the specified received Mailbag ID. (Users can determine the received Mailbag ID by viewing the receive status reports.) The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_mbag <mbagid>

Example 1

Requeue all files with RCV mbagid equal to 00054E.

```
ftp> quote mbx_rq_mbag 00054E
200-Requeue MBAG operation started.

     Requeue Successful; OUT810-1.000201.000301

     200 Requeue MBAG operation completed.
ftp>
```

Example 2

Requeue all files with RCV mbagid equal to 00055F.

```
ftp> quote mbx_rq_mbag 00055F
200-Requeue MBAG operation started.

     Requeue Successful; OUT810-2.000202.000302
     Requeue Successful; OUT810-3.000202.000303

   200 Requeue MBAG operation completed.
ftp>
```

Requeue Filename or Prefix—mbx_rq_file

Description

This command requeues all previously received mailbox files based on file name or file prefix. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_file <fn | file prefix>

Example 1

Requeue specific file.

```
ftp> quote mbx_rq_file  TDCC204-1.0000DI.000001976
200-Requeue FILE operation started.

    Requeue Successful; TDCC204-1.0000DI.000001976

200 Requeue FILE operation completed.
ftp>
```

Example 2

Requeue files based on file prefix.

```
ftp> quote mbx_rq_file EDIDATA
200-Requeue FILE operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970
    Requeue Successful; EDIDATA.0000DH.000001974

200 Requeue FILE operation completed.
ftp>
```

Display Valid Filename Prefixes—mbx_prefixes

Description

This command displays file name prefixes that the Advanced Data Distribution System will accept. You can configure the Advanced Data Distribution System to accept different file types from different Advanced Data Distribution users by setting up a separate *ftpaccess* file in a Advanced Data Distribution user's *user* directory. The parameter *mbx_qprefix* specifies which file types are allowed for the Advanced Data Distribution user.

Syntax

quote mbx_prefixes

Example

```
ftp> quote mbx_prefixes
200-Valid file prefixes for GENTRAN:Server Advanced Data Distribution System
are:
     EDI850
     edi850
     EDI837
     edi837

200 Done
ftp>
```

Report-Generating Commands

Dropped Off Status Report—mbx_rpt_sa

Description

This command generates a status report for all files dropped off. The report is generated for files located in users' *toco* and *came* directories. The generated report name is *snd.rpt*. Users can retrieve it using a **get** command.

Syntax

quote mbx_rpt_sa

Example

```
ftp> quote mbx_rpt_sa
200-Generating status report

DROPPED OFF STATUS REPORT
---------------------------------
Generated at 17:15:10 on 04/05/96

MbagID     Date Dropped Off    Date Processed     File Name

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

00011E     04/05/96  17:14     04/05/96 17:14     EDIDATA1.00011E.00016O
00011E     04/05/96  17:14     04/05/96 17:14     EDIDATA2.00011E.00016P
00011E     04/05/96  17:14     04/05/96 17:14     EDIDATA3.00011E.00016Q
00011E     04/05/96  17:14     04/05/96 17:14     EDIDATA4.00011E.00016R
00011E     04/05/96  17:14     04/05/96 17:14     EDIDATA5.00011E.00016S
00011E     04/05/96  17:15     04/05/96 17:15     TDCC204-1.00011E.00016T
00011E     04/05/96  17:15     04/05/96 17:15     TDCC204-2.00011E.00016U
00011E     04/05/96  17:15     04/05/96 17:15     TDCC204-3.00011E.00016V

Total number of files dropped off = 8
Total number of files processed = 8

200 Status report completed. Retrieve report snd.rpt via get command.
ftp>
```

Receive Status Report—mbx_rpt_ra

Description

This command generates a status report for all files that are picked up already or that are waiting to be picked up.

The report is generated for files located in users' *togo* and *gone* directories. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command.

Syntax

quote mbx_rpt_ra

Example

```
ftp> quote mbx_rpt_ra
200-Generating status report

RECEIVE STATUS REPORT
---------------------------------
Generated at 17:24:56 on 04/05/96


RCV

 M   MbagID     Ready For Pickup  MbagID    Date Received     File Name

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

 N   00011D     04/04/96  13:22                             OUTCLAIM1.00011D.000191
 N   00011D     04/04/96  13:22                             OUTCLAIM2.00011D.000192
 Y   00011E     04/05/96  17:23   00011F    04/05/96 17:24  OUTINV1.00011E.000193
 N   00011E     04/05/96  17:23   00011F    04/05/96 17:24  OUTINV2.00011E.000194
 N   00011C     04/04/96  13:22                             OUTRR1.00011C.000195
 N   00011C     04/04/96  13:22                             OUTRR2.00011C.000196

Total number of files = 6
Total number of files already picked up = 2
Total number of files waiting to be picked up = 4
Total number of marked files = 1

Note: 'M' - Marked files can not be requeued or received.

200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Waiting Status Report—mbx_rpt_rw

Description

This command generates a status report for files waiting to be picked up. The report is generated for files located in users' *togo* directory. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command.

Syntax

quote mbx_rpt_rw

Example

```
ftp> quote mbx_rpt_rw
200-Generating status report

WAITING STATUS REPORT
---------------------------------
Generated at 17:25:43 on 04/05/96

M   MbagID     Ready For Pickup          File Name

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

N   00011D     04/04/96  13:22           OUTCLAIM1.00011D.000191

N   00011D     04/04/96  13:22           OUTCLAIM2.00011D.000192

N   00011C     04/04/96  13:22           OUTRR1.00011C.000195

N   00011C     04/04/96  13:22           OUTRR2.00011C.000196

Total number of files = 4
Total number of files waiting to be picked up = 4
Total number of marked files = 0

Note: 'M' - Marked files can not be requeued or received.

200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Picked Up Status Report—mbx_rpt_rp

Description

This command generates a status report for files already picked up. The report is generated for files located in users' *gone* directory. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command. This report is used to determine which files to requeue.

Syntax

quote mbx_rpt_rp

Example

```
ftp> quote mbx_rpt_rp
200-Generating status report

PICKED UP STATUS REPORT
---------------------------------
Generated at 17:30:58 on 04/05/96


RCV

M   MbagID      Date Received   File Name

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

N   00011F      04/05/96 17:24  OUTINV1.00011E.000193

N   00011F      04/05/96 17:24  OUTINV2.00011E.000194


Total number of files picked up = 2
Total number of marked files = 0

Note: 'M' - Marked files can not be requeued or received.


200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Appendix

A

Command Reference Guide

Contents

Overview

Introduction

This appendix has been prepared for Advanced Data Distribution users' reference.

As per the following notices, licensees of GENTRAN:Server with the Advanced Data Distribution System are granted permission by Sterling Commerce to photocopy this appendix for distribution to their Advanced Data Distribution System users.

Notice

You may remove the following pages that make up the Command Reference Guide and photocopy them for distribution to your Advanced Data Distribution users.

User Maintenance Commands

Logging in to FTP—user

Log in and authentication

When users log in to obtain access to the Advanced Data Distribution System, several checks process to determine whether to grant them access to their account. During the login process, the FTP Daemon authenticates Advanced Data Distribution users' access privileges according to the criteria outlined in the following list.

- The user has set up a valid mailbox
- The user has entered a valid password.
- The password has not expired.
- The inactive period has not expired.
- Logins are enabled.
- Login attempts have not exceeded the *loginfails* value stored in the *ftpaccess* file.
- The Advanced Data Distribution machine is not denied access by the *ftphosts* file.
- The number of users in the Advanced Data Distribution user's class has not exceeded the *limit* value in the *ftpaccess* file.
- A schedule shutdown has not been established as defined by the *shutdown* file created by **ftpshut**.
- The parameters *edi_mbxru* and *EDI_ROOT* are defined in the *ftpaccess* file.

An informative message displays if the login process fails.

Starting FTP

To establish a connection between the host and the port number, enter the following FTP command:

ftp <IPAddress> <port#>

Where
<*IPAddress*> is the IP address of the machine where GENTRAN:Server Advanced Data Distribution resides (Line 4 on the Pre-Installation Checklist).

<*port#*> is the port number (Line 6 on the Pre-Installation Checklist).

(Continued on next page)

System Response
The system displays a message similar to the following:

Connected to 123.45.6.78
220 FTP server (Version wu-2.4(470) date time) ready.
Name(username):

Logging In

Users can log in to the FTP Daemon at any time. To access their account in a specific Advanced Data Distribution System, users use the **user** (Login User) command to log in to the FTP Daemon from any properly set up workstation.

Syntax

user <user name>

Log in process

The **user** command logs the Advanced Data Distribution user in and starts the mailbox session. This table describes the actions that occur during the log in process.

Stage	Description
1	If the user has an *ftpaccess* file in his or her *user* directory, the FTP Daemon reads this file after the default *ftpaccess* file.
2	The owner and group access of the FTP Daemon are determined by the pseudo Advanced Data Distribution UNIX account. The *edi_mbxru* parameter in the *ftpaccess* file specifies the pseudo UNIX login account. **Note** A UNIX account with a restricted access must exist for this step to occur.
3	The user has access to display the files in *./togo* directory of the mailbox.
4	A mailbag ID is generated for the current mailbox session.
5	The starting mailbox session message is recorded in the Mail Detail log.
6	The system displays the mailbox Help file containing valid mailbox commands.

(Continued on next page)

(Contd) Stage	Description
7	The system displays a welcome message.
8	The system displays a warning message if the password will expire within the number of days defined by the *passwd_warn_mes* parameter in the *ftpaccess* file.

Note
Due to variances in users' File Transfer Protocol configurations, the mailbox Help file and welcome message may not display on all systems. In order to see the message and file, users should use the quote mbx_help command.

Reference
See the Display Mailbox Help File—mbx_help topic later in this chapter for instruction on accessing the file.

Change Password—mbx_cngpw

Description

The change password command enables users to change their passwords. All activity is recorded in the Mail Detail log. The maximum length of the new password is nine characters.

A message similar to “200 ACCEPT - PASSWORD CHANGED “ is displayed if the password has been changed successfully, otherwise, an error message is displayed. A password change can fail for any of the following reasons:

- the old password was incorrect
- an error occurred reading the mailbox file
- the length of the new password exceeds nine characters

Syntax

quote mbx_cngpw <oldpasswd> <newpasswd>

Example 1

Change password from test to test2

```
ftp>  quote mbx_cngpw test test2
200 ACCEPT - PASSWORD CHANGED.
```

Example 2:

Unable to change password because length of *newpasswd* exceeds nine characters

```
ftp> quote mbx_cngpw test2 longnewpasswd
200-    Length of newpwd can not exceed 9 characters
200 REJECT - PASSWORD CHANGE FAILED.
```

Example 3:

Unable to change password because *oldpasswd* is incorrect

```
ftp> quote mbx_cngpw junk new
200- passw, oldpasswd incorrect
200 REJECT - PASSWORD CHANGE FAILED.
```

(Continued on next page)

Example 4: Unable to change password due to an error reading the file

```
ftp>  quote mbx_cngpw sadf asdf
200-    passw, abc_ifopen(mbox), iserrno = 2
200 REJECT - PASSWORD CHANGE FAILED.
```

Display Mailbox Help File—mbx_help

Description

This command displays the mailbox help file as defined in the *ftpaccess* file. Users whose File Transfer Protocol configuration impedes the automatic display of this file at login will use this command if they want to view the file.

Syntax

quote mbx_help

End Session—quit

Description

This command terminates the FTP session and exits. An end session record is placed in the Mail Detail log.

Syntax

quit

File Transfer and Requeuing Commands

Send One File—put

Description

The **put** command (Send One File) enables users to send a file to their mailbox.

Syntax

put <mailbox-file> [remote-file]

Using this syntax, you can send a particular mailbox file. If remote-file is unspecified, then the mailbox-file name is used as the remote-file name.

The put Process

This table describes the actions that occur when the **put** command is used.

Stage	Description
1	An error message is displayed if the user is trying to send a file outside of his or her mailbox directory.
2	The FTP Daemon determines whether to accept the user's file based on the file prefix. The FTP Daemon extracts the file prefix and compares it against all prefixes stored in the *ftpaccess* file. If a match is found, a copy of the file is sent from the FTP client to the Advanced Data Distribution System; otherwise the following message is displayed: fn: Permission denied. Not a valid file-prefix.
3	The file is renamed to <*filename*>.<*mailbagID*>.<*uniqueID*>.
4	The file is moved to the *toco* directory and registered as queued.
5	The file is routed to the Intelligent Agent's queue as defined by the mapping of file type to Intelligent Agent in the *ftpaccess* file.
6	The file is moved from the *toco* directory to the *came* directory and registered as dequeued.

(Continued on next page)

(Contd) Stage	Description
7	All activity is recorded in the Mail Detail log.
8	If the file was queued successfully to an Intelligent Agent, the following message is displayed: ********** * REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL * ********** otherwise, the following message is displayed: ********** * REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM FAILED * **********

Example 1 Send file EDIDATA1

```
ftp>  put EDIDATA1
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDIDATA1.00011F.000171.
226-Transfer complete.

**********
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********
226 Done.
236 bytes sent in 0.012 seconds (20 Kbytes/s)
```

(Continued on next page)

Example 2 Send file EDIDATA1 and store it as EDIDATA2 on remote host

```
ftp> put EDIDATA1 EDIDATA2
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDIDATA2.00011F.000172.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
236 bytes sent in 0.00098 seconds (2.4e+02 Kbytes/s)
```

Example 3 Unable to put files outside of their mailbox directory

```
ftp> put EDIDATA1 /etc/EDIDATA2
200 PORT command successful.
553-Permission denied. Unable to store file outside of advanced data distribution
system;
/etc/EDIDATA2

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM FAILED *
**********************************************************************************************
553 Done.
```

Example 4 Advanced Data Distribution System not set up to receive data file

```
ftp> put junk
200 PORT command successful.
553-junk: Permission denied. Not a valid file-prefix.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM FAILED *
**********************************************************************************************
553 Done.
```

Send Multiple Files—mput

Description

The **mput** (Send Multiple Files) command enables users to send multiple files to their mailboxes. The operations of this command are identical to the operations of the **put** (Send One File) command. All activity is recorded in the Mail Detail log.

Syntax

mput <*mailbox-file**>

The commands in the following examples send all mailbox files of a particular type.

Example 1

Send all files that begin with EDI850

```
ftp> mput EDI850*
local: EDI850-1 remote: EDI850-1
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDI850-1.00011H.00017E.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
local: EDI850-2 remote: EDI850-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      EDI850-2.00011H.00017F.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
```

(Continued on next page)

Example 2 Send all files that begin with TDCC204

```
ftp> mput TDCC204*
local: TDCC204-1 remote: TDCC204-1
200 PORT command successful.
150 Opening BINARY mode data connection for
      TDCC204-1.00011H.00017G.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
local: TDCC204-2 remote: TDCC204-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      TDCC204-2.00011H.00017H.
226-Transfer complete.

**********************************************************************************************
* REGISTRATION TO ADVANCED DATA DISTRIBUTION SYSTEM SUCCESSFUL *
**********************************************************************************************
226 Done.
36696 bytes sent in 0.0078 seconds (4.6e+03 Kbytes/s)
ftp>
```

Example 3 Send all mailbox files

```
mput *
```

Receive File—get

Description

The **get** (Receive File) command enables users to receive a file from their mailbox.

Syntax

get <mailbox-file> [local-file]

Using this syntax, users can retrieve a particular mailbox file. If local-file is unspecified, then the specified mailbox-file name is used as the local-file name.

This table describes the actions that occur when the **get** command is used.

Stage	Description
1	An error message is displayed if users try to retrieve files outside of their mailbox directory.
2	The FTP Daemon determines if the file is registered in the Advanced Data Distribution System. An error message displays if the file is not registered; otherwise, the FTP Daemon transfers a copy of the mailbox file to the user.
3	The mailbox file is moved from the user's *togo* directory to the *gone* directory and registered as dequeued.
4	All activity is recorded in the Mail Detail log.

Example 1

Retrieve file OUT856-1.00011V.000191.

```
ftp> get OUT856-1.00011V.000191
200 PORT command successful.
150 Opening BINARY mode data connection for
    OUT856-1.00011V.000191 (5 bytes).
226 Transfer complete.
5 bytes received in 0.00098 seconds (5 Kbytes/s)
```

(Continued on next page)

Example 2 Retrieve file OUT850-2.00011V.000189 and store it on the local host as /usr/users/data/OUT850-2.

```
ftp> get OUT850-2.00011V.000189 /usr/users/data/OUT850-2
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-2.00011V.000189 (5 bytes).
226 Transfer complete.
5 bytes received in 0.00098 seconds (5 Kbytes/s)
```

Example 3 Unable to retrieve file because it is not registered in the Advanced Data Distribution System.

```
ftp> get junk
200 PORT command successful.
550-    deque, isread, iserrno=111
550 DEQUEUE FAILED junk
```

Example 4 Unable to retrieve files outside of the user's mailbox.

```
ftp> get /usr/users/data/OUT850-2
200 PORT command successful.
550 Permission denied. Unable to retrieve file outside of
      mailbox system;
/usr/users/data/OUT850-2
```

Receive Multiple Files—mget

Description

The **mget** (Receive Multiple Files) command enables users to receive multiple files from their mailbox. The operations of this command are identical to the operations of the **get** command. All activity is recorded in the Mail Detail log.

Syntax

The following commands retrieve all mailbox files of a particular type.

mget <mailbox-file*>

Example 1

Retrieve all files that begin with OUT850.

```
ftp> mget OUT850*
local: OUT850-1.00011V.000188 remote:
        OUT850-1.00011V.000188
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-1.00011V.000188
(11 bytes).
226 Transfer complete.
11 bytes received in 0 seconds (0.011 Kbytes/s)
local: OUT850-2.00011V.000189 remote:
        OUT850-2.00011V.000189
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT850-2.00011V.000189
(5 bytes).
226 Transfer complete.
5 bytes received in 0.034 seconds (0.14 Kbytes/s)
```

(Continued on next page)

Example 2 Retrieve all files that begin with OUT856.

```
ftp> mget OUT856*
local: OUT856-1.00011V.000191 remote:
        OUT856-1.00011V.000191
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT856-1.00011V.000191
(5 bytes).
226 Transfer complete.
5 bytes received in 0 seconds (0.0049 Kbytes/s)
local: OUT856-2.00011V.000192 remote:
        OUT856-2.00011V.000192
200 PORT command successful.
150 Opening BINARY mode data connection for
      OUT856-2.00011V.000192
(5 bytes).
226 Transfer complete.
5 bytes received in 0 seconds (0.0049 Kbytes/s)
```

Example 3 Retrieve all mailbox files.

```
mget *
```

List Mailbox Files Waiting to be Picked Up—ls

Description

This command lists mailbox files waiting to be picked up. These files are located in the user's *togo* directory. Users are not allowed to view files outside of their mailbox directory. If users try to view a file not located in their mailbox directory, an error message displays.

Syntax

ls

Example 1

View all files waiting to be picked up.

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
OUT850-1.00011V.000188
OUT850-2.00011V.000189
OUT856-1.00011V.000191
OUT856-2.00011V.000192
rcv.rpt
226 Transfer complete.
```

Example 2

View (long format) all files waiting to be picked up.

Note
The filenames may continue onto the next lines differently on your screen display.

```
ftp> ls -l
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 40
-rw-rw-r--1 data    users          11 Apr 8 19:28 OUT850-1.00011V.000188
-rw-rw-r--1 data    users          5 Apr  8 18:59 OUT850-2.00011V.000189
-rw-rw-r--1 data    users          5 Apr  8 19:00 OUT856-1.00011V.000191
-rw-rw-r--1 data    users          5 Apr  8 19:00 OUT856-2.00011V.000192
-rw-rw-r--1 mbxusers users         646 Apr  8 19:29 rcv.rpt
226 Transfer complete.
```

(Continued on next page)

Example 3 Users are not allowed to view files outside of their mailbox directory.

```
ftp> ls /etc
200 PORT command successful.
550 Permission denied. Unable to view files outside of mailbox system; /etc
```

Requeue All Mailbox Files—mbx_rq_all

Description

This command requeues all previously received mailbox files. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_all

Example

In this example, the failed requeue occurred because the entry was marked.

```
ftp> quote mbx_rq_all
200-Requeue ALL operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Failed; can not requeue marked entry;
                          EDIDATA.00000E.000001014
    Requeue Successful; TDCC204-1.0000DI.000001976
    Requeue Successful; TDCC204-2.0000DI.000001978

200 Requeue ALL operation completed.
ftp>
```

Reference

For more information about marked entries, see the *GENTRAN:Server Advanced Data Distribution Guide*.

Requeue Files Received After Date—mbx_rq_after

Description

This command requeues all previously received mailbox files that were received after the specified date. The date is in the following format, full or partial: [CC]YYMMDDHHMM. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_after <date>

Example

```
ftp> quote mbx_rq_after 960331
200-Requeue AFTER operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Failed; can not requeue marked entry;
                          EDIDATA.00000E.000001014
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970

200 Requeue AFTER operation completed.
ftp>
```

Requeue Range—mbx_rq_range

Description

This command requeues all previously received mailbox files in the date range specified. The date is in the following format, full or partial: [CC]YYMMDDHHMM. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

```
quote mbx_rq_range <low date> <high date>
```

Example

```
ftp> quote mbx_rq_range 9604051646 960407
200-Requeue RANGE operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970
    Requeue Successful; EDIDATA.0000DH.000001974

200 Requeue RANGE operation completed.
ftp>
```

Requeue Mailbag Id—mbx_rq_mbag

Description

This command requeues all previously received mailbox files based on the specified received Mailbag ID. (Users can determine the received Mailbag ID by viewing the receive status reports.) The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_mbag <mbagid>

Example 1

Requeue all files with RCV mbagid equal to 00054E.

```
ftp> quote mbx_rq_mbag 00054E
200-Requeue MBAG operation started.

     Requeue Successful; OUT810-1.000201.000301

     200 Requeue MBAG operation completed.
ftp>
```

Example 2

Requeue all files with RCV mbagid equal to 00055F.

```
ftp> quote mbx_rq_mbag 00055F
200-Requeue MBAG operation started.

     Requeue Successful; OUT810-2.000202.000302
     Requeue Successful; OUT810-3.000202.000303

   200 Requeue MBAG operation completed.
ftp>
```

Requeue Filename or Prefix—mbx_rq_file

Description

This command requeues all previously received mailbox files based on file name or file prefix. The requeue operation moves the files from the users' *gone* to their *togo* directory. Users can retrieve the files once they are requeued. All activity is recorded in the Mail Detail log.

Syntax

quote mbx_rq_file <*fn* | *file prefix*>

Example 1

Requeue specific file.

```
ftp> quote mbx_rq_file  TDCC204-1.0000DI.000001976
200-Requeue FILE operation started.

    Requeue Successful; TDCC204-1.0000DI.000001976

200 Requeue FILE operation completed.
ftp>
```

Example 2

Requeue files based on file prefix.

```
ftp> quote mbx_rq_file EDIDATA
200-Requeue FILE operation started.

    Requeue Successful; EDIDATA.000009.000000997
    Requeue Successful; EDIDATA.0000DF.000001972
    Requeue Successful; EDIDATA.0000DG.000001970
    Requeue Successful; EDIDATA.0000DH.000001974

200 Requeue FILE operation completed.
ftp>
```

Display Valid Filename Prefixes—mbx_prefixes

Description

This command displays file name prefixes that the Advanced Data Distribution System will accept. You can configure the Advanced Data Distribution System to accept different file types from different Advanced Data Distribution users by setting up a separate *ftpaccess* file in a Advanced Data Distribution user's *user* directory. The parameter *mbx_qprefix* specifies which file types are allowed for the Advanced Data Distribution user.

Syntax

quote mbx_prefixes

Example

```
ftp> quote mbx_prefixes
200-Valid file prefixes for GENTRAN:Server Advanced Data Distribution System
are:
     EDI850
     edi850
     EDI837
     edi837

200 Done
ftp>
```

Report-Generating Commands

Dropped Off Status Report—mbx_rpt_sa

Description

This command generates a status report for all files dropped off. The report is generated for files located in users' *toco* and *came* directories. The generated report name is *snd.rpt*. Users can retrieve it using a **get** command.

Syntax

quote mbx_rpt_sa

Example

```
ftp> quote mbx_rpt_sa
200-Generating status report
DROPPED OFF STATUS REPORT
---------------------------------
Generated at 17:15:10 on 04/05/96
 MbagID     Date Dropped Off     Date Processed      File Name
 ------     ----------------     --------------      ----------
00011E     04/05/96  17:14     04/05/96 17:14      EDIDATA1.00011E.00016O
00011E     04/05/96  17:14     04/05/96 17:14      EDIDATA2.00011E.00016P
00011E     04/05/96  17:14     04/05/96 17:14      EDIDATA3.00011E.00016Q
00011E     04/05/96  17:14     04/05/96 17:14      EDIDATA4.00011E.00016R
00011E     04/05/96  17:14     04/05/96 17:14      EDIDATA5.00011E.00016S
00011E     04/05/96  17:15     04/05/96 17:15      TDCC204-1.00011E.00016T
00011E     04/05/96  17:15     04/05/96 17:15      TDCC204-2.00011E.00016U
00011E     04/05/96  17:15     04/05/96 17:15      TDCC204-3.00011E.00016V
Total number of files dropped off = 8
Total number of files processed = 8

200 Status report completed. Retrieve report snd.rpt via get command.
ftp>
```

Receive Status Report—mbx_rpt_ra

Description

This command generates a status report for all files that are picked up already or that are waiting to be picked up.

The report is generated for files located in users' *togo* and *gone* directories. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command.

Syntax

quote mbx_rpt_ra

Example

```
ftp> quote mbx_rpt_ra
200-Generating status report
RECEIVE STATUS REPORT
---------------------------------
Generated at 17:24:56 on 04/05/96

RCV
```

M	MbagID	Ready For Pickup	MbagID	Date Received	File Name
--	------	----------	------	----------	----------
N	00011D	04/04/96 13:22			OUTCLAIM1.00011D.000191
N	00011D	04/04/96 13:22			OUTCLAIM2.00011D.000192
Y	00011E	04/05/96 17:23	00011F	04/05/96 17:24	OUTINV1.00011E.000193
N	00011E	04/05/96 17:23	00011F	04/05/96 17:24	OUTINV2.00011E.000194
N	00011C	04/04/96 13:22			OUTRR1.00011C.000195
N	00011C	04/04/96 13:22			OUTRR2.00011C.000196

```
Total number of files = 6
Total number of files already picked up = 2
Total number of files waiting to be picked up = 4
Total number of marked files = 1

Note: 'M' - Marked files can not be requeued or received.

200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Waiting Status Report—mbx_rpt_rw

Description

This command generates a status report for files waiting to be picked up. The report is generated for files located in users' *togo* directory. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command.

Syntax

```
quote mbx_rpt_rw
```

Example

```
ftp> quote mbx_rpt_rw
200-Generating status report
WAITING STATUS REPORT
---------------------------------
Generated at 17:25:43 on 04/05/96

M    MbagID     Ready For Pickup    File Name
--   ------     ----------------    ---------
N    00011D     04/04/96  13:22     OUTCLAIM1.00011D.000191
N    00011D     04/04/96  13:22     OUTCLAIM2.00011D.000192
N    00011C     04/04/96  13:22     OUTRR1.00011C.000195
N    00011C     04/04/96  13:22     OUTRR2.00011C.000196

Total number of files = 4
Total number of files waiting to be picked up = 4
Total number of marked files = 0

Note: 'M' - Marked files can not be requeued or received.

200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Picked Up Status Report—mbx_rpt_rp

Description

This command generates a status report for files already picked up. The report is generated for files located in users' *gone* directory. The generated report name is *rcv.rpt*. Users can retrieve it using a **get** command. This report is used to determine which files to requeue.

Syntax

quote mbx_rpt_rp

Example

```
ftp> quote mbx_rpt_rp
200-Generating status report
PICKED UP STATUS REPORT
---------------------------------
Generated at 17:30:58 on 04/05/96

RCV
M       MbagID      Date Received     File Name
--      ------      -------------     -------------
N       00011F      04/05/96 17:24    OUTINV1.00011E.000193
N       00011F      04/05/96 17:24    OUTINV2.00011E.000194

Total number of files picked up = 2
Total number of marked files = 0

Note: 'M' - Marked files can not be requeued or received.

200 Status report completed. Retrieve report rcv.rpt via get command.
ftp>
```

Index

Symbols

A

C

D

E

F

G

I

L

M