English

FUJITSU Software

openFT

Concepts and Functions

User Guide

Valid for:
openFT V12.1C40

November 2022

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to: bs2000services@ts.fujitsu.com.

Certified documentation according to DIN EN ISO 9001:2015

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2015.

Copyright and Trademarks

Table of Contents

Concepts and Functions **8**
1 Preface **9**
1.1 Brief description of the product **10**
1.2 Target group **11**
1.3 Concept of openFT manuals **12**
1.4 Changes since the last version **14**
1.4.1 Changes for all platforms 15
1.4.2 Changes for Unix and Windows platforms 18
1.4.3 Changes for Unix platforms 19
1.4.4 Changes for BS2000 systems and z/OS 20
1.4.5 Changes for z/OS 21
1.4.6 New functions that are only available in the openFT Explorer 22
1.5 Notational conventions **23**
1.6 Internet **24**
2 openFT - the Managed File Transfer **25**
2.1 Heterogeneous computer systems **27**
2.1.1 File conversion 28
2.1.2 openFT product range 29
2.2 Heterogeneous networks **31**
2.2.1 The OSI reference model 32
2.2.2 Position of the openFT product family in the OSI Reference Model 34
2.2.3 openFT partners 35
2.2.4 FTAM partners 36
2.2.5 FTP partners 37
2.3 Transferring files **38**
2.3.1 Specifying the transfer start time 39
2.3.2 Controlling the duration of a request 40
2.3.3 Request queue 41
2.3.4 Automatic restart 42
2.4 File management **43**
2.5 Remote command execution **44**
2.6 Automation **45**
2.6.1 File transfer with preprocessing, postprocessing and follow-up processing . . 46
2.6.1.1 Preprocessing and postprocessing 47
2.6.1.2 Follow-up processing 48
2.6.2 Program interfaces 49
2.6.3 openFT script interface on Unix and Windows systems 50

2.6.4 Job variables in BS2000 systems 51
2.6.5 Automated further processing of openFT data 52
2.7 Secure operation 53
2.7.1 The FTAC function 54
2.7.1.1 Features of the FTAC function 55
2.7.1.2 Admission set 57
2.7.1.3 Admission profile (FT profile) 58
2.7.2 Encryption for file transfer requests and file management requests 60
2.7.3 Logging openFT operations - the logging function 61
2.7.4 Authentication 62
2.8 Using openFT in a cluster or HIPLEX/Sysplex composite 63
3 Partner concept 64
3.1 Partner list 65
3.1.1 Partner addresses 66
3.1.2 Partner types 67
3.2 FTAC security levels for partner entries 69
3.3 Outbound and inbound deactivation of named partners 70
3.4 Serialization of asynchronous outbound requests 71
3.5 Prioritization of partners 72
3.6 Using PCMX and TNS in Unix and Windows systems 73
4 File transfer and file management 74
4.1 File names 75
4.1.1 Unique file names for receive files 76
4.1.2 BS2000 file names 77
4.1.3 File names in Unix systems 79
4.1.4 Windows file names 80
4.1.5 z/OS file names 81
4.2 File passwords 85
4.3 File types 86
4.3.1 BS2000 files 87
4.3.2 z/OS files 88
4.3.3 Unix and Windows files 89
4.3.4 FTAM files 91
4.3.5 Transfer of various file types 92
4.3.6 Transferring directories (Unix and Windows systems) 97
4.3.7 Migrated files 98
4.4 Transferring 7-bit, 8-bit and Unicode files 99
4.4.1 Code tables and coded character sets (CCS) 100
4.4.2 Specifying the CCS on a transfer request 101
4.4.3 Data conversion 102
4.4.4 XHCS support by openFT on BS2000 systems 104

4.5 Character code support for file names and commands on Unix and Windows systems 105
4.5.1 Properties of the character mode 106
4.5.2 Recommendations for the use of transparent and character mode 107
4.5.2.1 Using the transparent mode 108
4.5.2.2 Using the character mode 109
4.6 Entries for the remote system 111
4.6.1 Defining the partner computer 112
4.6.2 Transfer admission 113
4.7 Options for file transfer 114
4.7.1 Maximum record lengths 115
4.7.2 Write mode 116
4.7.3 Compressed file transfer 117
4.7.4 Encrypted file transfer and encrypted file management 118
4.7.5 Transfer of protection attributes between BS2000 systems 119
4.7.6 Notifying results 120
4.7.7 Access mode for FTAM partners 121
4.7.8 Preprocessing and postprocessing 122
4.7.9 Follow-up processing 124
4.8 Access protection for send and receive files 125
4.8.1 Access protection during transfer 126
4.8.2 Specific features in z/OS 128
4.9 File management 129
4.9.1 File management in the remote system 130
4.9.2 File management in the local system 131
4.10 Special points for file transfer with FTAM partners 133
4.10.1 Virtual filestore 134
4.10.2 Addressing via Application Entity Title (AET) 137
4.10.2.1 Addressing FTAM partners with AET in Object Identifier Form 138
4.10.2.2 Extended support of the Application Entity Title on Unix and Windows systems 140
4.10.3 Synchronous transfer of multiple files with FTAM 141
5 Security functions 142
5.1 FTAC functions 143
5.1.1 Admission sets 144
5.1.2 Admission profiles 146
5.1.3 The FTAC logging function 149
5.2 Authentication 150
5.2.1 Authentication usages 151
5.2.2 Instance identifications 152
5.2.3 Local RSA key pairs 153

5.2.3.1 Key pair attributes . . . 154
5.2.3.2 Updating key files and replacing keys . . . 155
5.2.3.3 Imported keys . . . 156
5.2.4 Keys of partner systems . . . 157
5.2.5 Secure distributing of keys . . . 158
5.3 Extended authentication check . . . 159
5.4 Encryption for file transfer and file management . . . 160
5.4.1 Encryption of the request description data . . . 161
5.4.2 Encryption of the file contents . . . 162
5.4.3 Encryption for file management requests . . . 163
5.5 Protection mechanisms against data manipulation . . . 164
5.6 Encryption with FTPS . . . 165
6 Working with openFT . . . 166
6.1 Command interface . . . 167
6.2 openFT Explorer for Unix and Windows systems . . . 168
6.3 Program interface . . . 169
6.4 openFT-Script interface . . . 170
6.5 ISPF panels for z/OS . . . 171
7 Administration . . . 172
7.1 Role concept for administration . . . 173
7.2 Local administration . . . 175
7.2.1 Starting and stopping openFT . . . 176
7.2.2 Administering openFT operating parameters . . . 177
7.2.3 Administering RSA keys . . . 178
7.2.4 Monitoring and controlling openFT operation . . . 179
7.2.4.1 Administering requests . . . 180
7.2.4.2 Administering FT logging . . . 181
7.2.4.3 Administration via SMNP . . . 182
7.2.4.4 Controlling the FT monitoring function . . . 183
7.2.4.5 Evaluating console messages . . . 184
7.2.4.6 Evaluating ADM traps . . . 185
7.2.4.7 Other administration functions . . . 186
7.2.5 Administering partners . . . 187
7.2.6 Diagnostics and error correction . . . 188
7.2.7 Administering admission sets and admission profiles . . . 189
7.2.7.1 Defining the standard admission set . . . 190
7.2.7.2 Defining and maintaining individual admission sets . . . 191
7.2.7.3 Administering admission profiles . . . 192
7.2.7.4 Transfer FTAC environment - the environment functions . . . 193
7.3 Remote administration via openFT Explorer . . . 194
7.4 Central administration . . . 195

7.4.1 Remote administration 196
7.4.1.1 The remote administration concept 197
7.4.1.2 Configuration with multiple remote administration servers 199
7.4.2 ADM traps 201
8 License provisions 202
9 Glossary 206
10 Abbreviations 232
11 Related publications 238

Concepts and Functions

1 Preface

The openFT product range transfers and manages files

- automatically,
- securely, and
- cost-effectively.

The reliable and user-friendly transfer of files is an important function in a high-performance computer network. The corporate topologies consist of networked PC workstations, which are usually additionally linked to a mainframe or Unix based server or Windows server. This allows much of the processing power to be provided directly at the workstation, while file transfer moves the data to the mainframe for further processing there as required. In such landscapes, the locations of the individual systems may be quite far apart. Fujitsu offers an extensive range of file transfer products - the openFT product range - for the following system platforms:

- BS2000®
- Linux® (Intel x86_64)
- Microsoft® Windows™ 10, Windows Server 2016 and 2019
- z/OS (IBM®)

1.1 Brief description of the product

The openFT product range comprises the following products:

FUJITSU Software openFT (Unix systems) is the file transfer product for systems with a Unix based operating system.

FUJITSU Software openFT (Windows) is the file transfer product for Microsoft's Windows systems.

FUJITSU Software openFT (BS2000) is the file transfer product for computers using the operating system BS2000.

FUJITSU Software openFT (z/OS) is the file transfer product for computers using the operating system z/OS.

All openFT products communicate with each other using the openFT protocol (previously only known as FTNEA) as laid down by Fujitsu. Since a number of FT products from other software vendors also support these protocols, many interconnection options are available.

openFT allows the use of the following transport protocols:

- TCP/IP
- ISO TP0/2 (not on z/OS)
- ISO TP4 (not on z/OS)
- SNA (only on z/OS)

The range of functions made available by openFT can be extended by:

- FTAC:

  FTAC provides extended system and data access protection. FTAC stands for File Transfer Access Control.
  On BS2000 systems, FTAC is provided by the add-on product openFT-AC.
  On z/OS, FTAC is provided by the add-on product openFT-AC.
  On BS2000 systems and on z/OS, FTAC is provided by the add-on product openFT-AC.
  On Unix and Windows systems, FTAC is integrated in openFT.
- openFT-FTAM (not available on z/OS):

  openFT supports the FTAM file transfer protocol (File Transfer Access and Management) standardized by ISO (International Organization for Standardization). This makes it possible to interconnect with even more systems from other vendors whose file transfer products support the same standard.
- openFT-FTP:

  openFT also supports the FTP functionality. This makes it possible to interconnect with other FTP servers.
- openFT-CR:
  openFT-CR is required up to version V12.1B if encrypted file transfer is requested.

> **Attention**
> As of openFT version V12.1C, openFT-CR is no longer delivered because the functionality has been integrated in openFT V12.1C.
>
> All references to openFT-CR therefore only apply to openFT versions <= V12.1B.

1.2 Target group

This manual, “Concepts and Functions” introduces you to the openFT product family and enables you to get started working with openFT. It is aimed particularly at those who are not yet familiar with openFT. But even if you already know and work with openFT, you will find the manual useful for the overview it provides of the product’s range of functions and capabilities.

Rather than being concerned with the syntactic subtleties of individual statements or with the details of specific interfaces, the manual provides a general overview of the features and possible applications of openFT. Equipped with this information, you will have no problem understanding the other manuals in the openFT series.

1.3 Concept of openFT manuals

openFT - Concepts and Functions

This manual is intended for those who want to get familiar with the capabilities of openFT and want to understand the openFT functions. It describes:

- the concept of openFT as a Managed File Transfer
- the scope of work and main features of the openFT product family
- the openFT-specific terms

openFT (Unix and Windows Systems) - Installation and Operation

This manual is intended for the FT, FTAC and ADM administrator on Unix and Windows systems. It describes:

- how to install openFT and its optional components
- how to operate, control and monitor the FT system and the FTAC environment
- the configuration and operation of a remote administration server and a ADM trap Server

openFT (BS2000) - Installation and Operation

This manual is intended for the FT and FTAC administrator on BS2000 systems. It describes:

- how to install openFT and its optional components on the BS2000 system
- how to operate, control and monitor the FT system and the FTAC environment
- the accounting records

openFT (z/OS) - Installation and Operation

This manual is intended for the FT and FTAC administrator on z/OS. It describes:

- how to install openFT and its optional components, including the requirements for using the product
- how to operate, control and monitor the FT system and the FTAC environment
- the openFT and openFT-AC messages for the FT administrator
- additional sources of information for the FT administrator, such as the accounting records and the logging information

openFT (Unix and Windows Systems) - Command Interface

This manual is intended for the openFT users on Unix and Windows systems and describes:

- the conventions for file transfer to computers with different operating systems
- the openFT commands on Unix and Windows systems
- the messages of the various components

The description of the openFT commands also applies to the POSIX interface on BS2000 systems.

openFT (BS2000) - Command Interface

This manual is intended for the openFT users on BS2000 systems and describes:

- the conventions for file transfer to computers with different operating systems

- the openFT commands on BS2000 systems
- the messages of the various components

openFT (z/OS) - Command Interface

This manual is intended for the openFT users on z/OS systems and describes:

- the conventions for file transfer to computers with different operating systems
- the openFT commands on z/OS
- the menu interface for the FT administrator and the FT user
- the program interface for the FT user
- the messages of the various components

openFT (BS2000) - Program Interface

This manual is intended for the openFT programmer and describes the openFT and openFT-AC program interfaces on BS2000 systems.

openFT (Unix and Windows Systems) - C and Java Program Interface

This manual is intended for C and Java programmers on Unix and Windows systems. It describes the C program interface and the main features of the Java interface.

openFT (Unix and Windows Systems) - openFT-Script Interface

This manual is intended for XML programmers and describes the XML statements for the openFT-Script interface.

> Many of the functions described in the manuals can also be executed via the openFT graphical interface, the openFT Explorer. The openFT Explorer is available on Unix systems and Windows systems. You can use the openFT Explorer to operate, control and monitor the FT system and the FTAC environment of remote openFT installations on any system platform independent from the local system, A detailed online help system that describes the operation of all the dialogs is supplied together with the openFT Explorer.

1.4 Changes since the last version

This section describes the changes in openFT V12.1 compared to openFT V12.0A.

> The functional extensions to the openFT commands, whether they relate to administrators or users, are also available in the openFT Explorer which is provided on Unix and Windows systems. For details, see the *New functions* section in the associated online help system.
> On z/OS, the functional extensions are also available in the menu system (panels).

1.4.1 Changes for all platforms

- Extended Unicode support

  On all Unicode capable systems, file names, FTAC transfer admissions and follow-up processing may consist of Unicode characters. To permit this, the function "Encoding Mode" has been introduced in order to represent the Unicode names correctly on all involved systems.

  The command interfaces have been extended as follows:

  - All platforms:

    The new field FNC-MODE in the long output of log records displays the encoding mode for the file name (commands *ftshwl*, SHOW-FT-LOGGING-RECORDS and FTSHWLOG). On BS2000 systems, the OPS variables have been extended by the elements FNC-MODE and FNCCS.
  - BS2000 and z/OS systems (from openFT V12.1C): the new option FILE-NAME-ENCODING supports different character encoding for the specification of remote file names, pre, post and follow-up processing for transfer commands for the following commands:
    - TRANSFER-FILE-SYNCHRONOUS / FTSCOPY
    - TRANSFER-FILE / FTACOPY
    - CREATE-REMOTE-DIR / FTCREDIR
    - DELETE-REMOTE-DIR / FTDELDIR
    - MODIFY-REMOTE-DIR-ATTRIBUTES / FTMODDIR
    - MODIFY-REMOTE-FILE-ATTRIBUTES / FTMOD
    - SHOW-REMOTE-FILE-ATTRIBUTES / FTSHW
    - DELETE-REMOTE-FILE / FTDEL
    - EXECUTE-REMOTE-CMD / FTEXEC
    - EXECUTE-REMOTE-FTADM-CMD / FTADM.
  - Unix systems and Windows systems:
    - New option *-fnc* in order to set the encoding mode in a file transfer, file management or administration request. This option is available for the commands *ft*, *ftadm*, *ftcredir*, *ftdel*, *ftdeldir*, *ftexec*, *ftmod*, *ftmoddir*, *ftshw* and *nopy*.The encoding mode is displayed in the output of the following commands (in addition to *ftshwl*): *ftshw* and *ftshwr* (FNC-MODE field).
      The number of not mapped file names is displayed using *ftshw -sif*.
    - New attribute *CmdMode* in the configuration of remote administration server to define the (recommended) encoding mode for administered openFT instances. The encoding mode is displayed in the output of the *ftshwc* command (MODE field).
      This function is also available in the configuration editor of the openFT Explorer.
    - In Unix systems, it is also possible to set the character set which is to be used for inbound requests in character mode. To do this, the new option *-fnccs* in the *ftmodo* command has been introduced.

      The character set which is currently set for inbound requests in character mode is displayed in *ftshwo*, FN-CCS-NAME field.
    - Unicode for C and JAVA

- BS2000 and z/OS systems: for library elements the log records display the element name, type and version in addition to the library name .

- For inbound requests, the long output and CSV output of log records display the address of the partner system in the new field PTNR-ADDR. On BS2000 systems, the partner address is also displayed in the OPS variable PARTNER-ADDRESS.
- Deactivation of the restart functions

  The restart function can be deactivated for asynchronous file transfer requests via the openFT or FTAM protocol. The restart can be set partner-specifically for outbound requests and globally for inbound and outbound requests. To permit this, the following commands have been modified:

  Unix and Windows systems:

  - *ftaddptn* and *ftmodptn*: New option *-rco*
  - *ftmodo*: New options *-rco* and *-rci*

  BS2000 and z/OS systems:

  - ADD-FT-PARTNER/MODIFY-FT-PARTNER and FTADDPTN/FTMODPTN:
    New operand RECOVERY-OUTBOUND
  - MODIFY-FT-OPTIONS and FTMODOPT:
    New operands RECOVERY-OUTBOUND and RECOVERY-INBOUND

- Minimum RSA key length for openFT protocol

  An openFT instance can require a minimum RSA key length for the openFT session encryption. The minimum RSA key length can be defined in the operating parameters. To permit this, the following commands have been modified:

  Unix and Windows systems:

  - *ftmodo*: New option *-klmin*

  BS2000 and z/OS systems:

  - MODIFY-FT-OPTIONS and FTMODOPT: New parameters RSA-PROPOSED and RSA-MINIMUM for the KEY-LENGTH operand.

- Minimum AES key length for openFT protocol

  An openFT instance can require a minimum AES key length for the openFT session encryption. The minimum AES key length can be defined in the operating parameters. To permit this, the following commands have been modified:

  Unix and Windows systems:

  - *ftmodo*: New option *-aesmin*

  BS2000 and z/OS systems:

  - MODIFY-FT-OPTIONS and FTMODOPT: New parameter AES-MINIMUM for the KEY-LENGTH operand.

- Encryption for file management requests

  For file management requests the file and directory list attributes can be transferred encrypted..This property can also set in admission profiles.

  To permit this, the following commands have been modified:

  Unix and Windows systems:

  - *ftshw*: New option *-c*
  - *ftcrep* and *ftmodp*: New option *-cm*
  - *ftshwp*: New parameter FILE-AT-ENC in the long output and new parameter *FileAtEnc* in the csv output.

  BS2000 and z/OS systems:

  - SHOW-REMOTE-FILE -ATTRIBUTES and FTSHW:
    New operand FILE-ATTR-ENCRYPTION
  - CREATE-FT-PROFILE and MODIFY-FT-PROFILE as well as FTCREPF and FTMODPRF: New operand FILE-ATTR-ENCRYPTION
  - SHOW-FT-PROFILE and FTSHWPRF: New parameter FILE-AT-ENC in the long output and new parameter *FileAtEnc* in the csv output.

  For getting multiple files form the remote system using ft_mget, GET-REMOTE-FILES or FTMGET the encryption option (-c or DATE-ENCRPTION) also applies to file and directory list attributes.
- From openFT V12.1C the CRYPT functionallity is integrated in openFT. Therefore openFT-CR is no longer delivered.

Changes in openFT version V12.1C10:

- For local outbound requests, the long output and CSV output of log records display the remote filename in the new field REMOTE-FN. On BS2000 systems, the remote filename is also displayed in the OPS variable F-REM-NAME.

1.4.2 Changes for Unix and Windows platforms

- Transferring directories:
  - Directories can be transferred between Unix and Windows systems. To permit this, the commands *ft* and *ncopy* have been extended with the option *-d*.
  - The new field PROGRESS in the output of the *ftshwr* command displays the progress of (asynchronous) directory transfer.
  - The new option *ftmodo -ltd* has been introduced to set the logging scope for directory transfer.
  - The new value *ftshwl -ff=T* selects log records of directory transfer requests. In addition, the *ftshwl* output has been extended to the field TRANSFILE (long output) as well as the FT function values TD, SD, SF (short output) and the value FUNCTION=TRANSFER-DIR (long output).
- Transferring multiple files via FTAM:

  Multiple files can be transferred synchronously between Unix and Windows systems using the FTAM protocol. This is controlled by a specific file name syntax of the *ncopy* command.
- Extension of the openFT-Script commands
  - The FT administrator can set limits of openFT requests. To permit this, the command *ftmodsuo* has been extended to the options *-u*, *-thl* and *-ftl*.
  - *ftshwsuo* displays the limits currently set.
- The *ftshwk* command displays the partner name for public keys of partner systems.
- FarSync X25 support

  FarSync X.25 cards from the manufacturer FarSite are directly supported by openFT on Linux and Windows systems. PCMX is no longer required for this. The connection method XOT (X.25 via TCP/IP) is also supported on Linux by using the FarSync XOT Runtime.
  To permit this, the commands *ftaddptn*, *ftmodptn*, *ftmodo*, *ftshwptn* and *ftshwo* have been extended.
- Extended support of the Application Entity Title

  The Application Entity Title (AET) now can be used for checking the partner address of FTAM partners. To permit this, the *ftmodo* command has been modified by extending the *-ptc* (partner check) option and adding the *-aet* option for specifying the AET. The *ftshwo* command has been extended by the *-ae* option.
- The maximal length of the command string in the *ftexec* command has been raised to 8191.
- Other changes
  - Modified partner checking for partners which are addressed via IPv6 with scope ID or via X.25 with line number. By this, a unique identification via the partner address is always possible.
  - The *ft_mget* command has been extended by the *-case* option which controls the consideration of the upper case / lower case in the file name pattern.
  - The ADM administrator now can return the permission for remote administration (*ftmoda -admpriv=n*). The configuration of the remote administration server is retained.

Changes in openFT version V12.1C20:

- Support of RSA keys with length 3072 and 4096.

Changes in openFT version V12.1C30:

- On Windows systems the openFT administrator rights can be transferred to any user.
  The *ftmodo* command has been extended for this purpose.

1.4.3 Changes for Unix platforms

- Single-user mode

  On Unix systems, the administrator can switch between the multi-user mode (default) and the single-user mode using the *ftsetmode* command. In single-user mode openFT runs completely under a specific user ID (the so called openFT ID) which is also FT and FTAC administrator. To permit creating and administering additional openFT instances in single-user mode, the commands *ftcrei* and *ftmodi* have been extended by the option *-ua* for specifying the user ID of a new instance.
- SNMP is no longer supported on Unix platforms.
- Support of systemd.

Changes in openFT version V12.1C10:

- Support of Openshift.

Changes in openFT version V12.1C30:

- Installation using license keys.

Changes in openFT version V12.1C40:

- On Unix systems, as already in Windows systems, the openFT administrator rights can be transferred to any user.
  The *ftmodo* command has been extended for this purpose.

1.4.4 Changes for BS2000 systems and z/OS

- New commands GET-REMOTE-FILES (BS2000 systems) and FTMGET (z/OS) for synchronous or asynchronous fetching of multiple files specified by wildcards from a remote system.
- New diagnostics command FTPING on BS2000-POSIX and z/OS for testing the openFT connection to a remote partner.

1.4.5 Changes for z/OS

- The PARM member of the z/OS parameter file has been changed as follows:
  - New key word JOB_JOBCLASS for follow-up processing jobs, preprocessing jobs, postprocessing jobs and print jobs.
  - New key word LISTPARM for setting of a default printer (LISTING=*STD in a FT request).
  - The key word JOB_MSGCLASS now applies to preprocessing jobs and postprocessing jobs.
- For FJBATCH in z/OS as of V2.1, you can use the PARMDD parameter instead for the PARM parameter.
- NCOPY and FTACOPY: New value LISTING=*STD in LOCAL-PARAMETER in order to assign a printer defined via LISTPARM.
- openFT (z/OS) is now supporting host names with up to 80 characters in length. This applies both to the internal communication in z/OS and to connections to z/OS partners.
- The member TNCTCPIP of the z/OS parameter file is no longer supported, therefore the description has been dropped.

Changes in openFT version V12.1C30:

- The PARM member of the z/OS parameter file has been changed as follows:
  - New key word TZSTRING for support of timezone handling to avoid missing lines when reading logging records that were written during the change from daylight saving time to winter time.

1.4.6 New functions that are only available in the openFT Explorer

- Exporting public keys

  The FT administrator can export public keys of the local openFT instance using the *Key Management* - *Export Public Key* command in the *Administration* menu.

- Deleting diagnosis information and console messages

  The FT administrator can delete diagnosis information and console messages using the commands *Delete Diagnosis Information* and *Delete Console Messages* in the *Administration* menu.

- The logging is also available in the object tree of the openFT Explorer.

Please refer to the online help for more details.

1.5 Notational conventions

The following notational conventions are used throughout this manual:

`typewriter font`

typewriter font is used to identify entries and examples.

indicates notes.

Indicates warnings.

1.6 Internet

Current information on the Internet

Current information on the openFT family of products can be found in the internet under http://www.fujitsu.com/ts/openFT.

2 openFT - the Managed File Transfer

Managed File Transfer is a term that documents the high performance of openFT products. Such high demands on corporate file transfer result, on the one hand, from the variety of hardware and software commonly installed today and, on the other, from the different needs your company has with respect to file transfer itself. A further important aspect of enterprise file transfer is provided by the options for automation and the security functions offered by openFT. In addition, central administration of an openFT network and presentation of the operating states make openFT a managed file transfer system.

Fujitsu Technology Solutions offers a comprehensive openFT product range for Managed File Transfer, which can be used to operate **heterogeneous computer systems** (hardware and software) of many manufacturers ranging from mainframe systems to the PC. openFT products can be used in various operating systems such as Windows, Unix systems, BS2000 systems, z/OS and others.

Even **heterogeneous networks** such as TCP/IP, ISO-FTAM, X.21/X.25, ISDN and GSM mobile telephony or MODACOM pose no problem for openFT. The continual integration of new platforms and network types guarantees high availability of the openFT products, also in the future. Not all networks are supported on all platforms.

The integration of the **ISO 8571 FTAM standard** (File Transfer, Access and Management) guarantees uniform interfaces for requests to openFT partners and any FTAM partners (not available under z/OS).

Support for the **FTP protocol** makes it possible to connect to FTP servers and FTP clients on any required platform.

Functions such as request storage, automatic restart, job and file management, follow-up processing, resource management, program interfaces, encryption and authentication indicate the wide range of services offered by openFT products, thus making them truly suitable for Managed FIle Transfer.

Request storage makes it possible to start **asynchronous file transfer** at any desired time, e.g., to save charges or to wait for the occurrence of specific events. The **automatic restart** feature ensures a consistent continuation of file transfer after the correction of a fault, e.g., a network or processor failure.

Automation is achieved, among other things, via facilities for preprocessing and follow-up processing:

- Local or remote **preprocessing** enables data to be created within a send or receive request by starting a job, for example, and then transferring it then to the local or remote system.
- Local or remote **postprocessing** enables the data transferred to be processed further within a send or receive request.
- Preprocessing as well as postprocessing can be executed within a request.
- **Follow-up processing** permits any job to be started just after file transfer. You can make the start of follow-up processing dependent on the success of the file transfer.

The **program interfaces** permit the implementation of openFT functions in programs.

File management in the remote and local systems allows you to modify file attributes, for example.

The **resource control** allows you to store file transfer requests at any time and have them issued automatically when the partner system is available. The use of Monitor Job Variables in BS2000 systems is also possible.

In the case of **synchronous file transfer**, you have to wait until data transfer has been completed and you can then react immediately to the result.

Protection of the data inventory is becoming a priority issue in companies in view of the open nature of today's networks. The **FTAC functionality** (optional in openFT (BS2000) and openFT (z/OS)) integrated in openFT products offers comprehensive and individually scalable protection functions:

- decoupling of transfer admissions and login authorization
- access rights dependent on the partner systems
- user-specific access rights
- flexible access right levels
- recording of every authorization check

The **logging** of data transfer requests and authorization checks permits evaluation of previous request and access, thus providing a further security feature.

The **encryption** of request description and transfer data is another protection level provided by openFT. Request description data include the authorization data for the transfer of and access to data (e.g. transfer admission, file password). In addition, it is possible to connect to system security functions such as SECOS on BS2000, RACF and ACF2 on z/OS.

Expanded identity checking (i.e. **authentication**) of the communications partner is offered for requests involving openFT partners. It is based on addressing network-wide, unique IDs for openFT instances and the exchange of partner-specific key information.

In the case of very large numbers of files (e.g. entire directory trees), the openFT-Script interface supports **restartable transfer**, i.e. if the network or the computer crashes on the 258th file, openFT-Script resumes the transfer at precisely this point following the restart. The openFT-Script interface is available on Unix systems and Windows systems.

2.1 Heterogeneous computer systems

One strength of the openFT products is their capability for linking different computers, particularly computers from different manufacturers running various operating systems. The precondition for file transfer between two computers is that a transport connection exists between these two computers and that one of the openFT products, an FTAM product or an FTP application is installed on the computers. FTAM is not available on z/OS.

The openFT products are matched for optimum interoperability. They retain file structures and attributes during file transfer. openFT products cannot override the conventions that apply to the operating system. Data conversion may be necessary to ensure that characters are represented correctly when performing transfers between certain operating systems.

2.1.1 File conversion

The coding, i.e. the system-internal representation of individual characters, letters and digits, depends on the operating system. The data must then be converted because

- Internally, Unix and Windows computers use an ASCII-based code (American Standard Code for Information Interchange). For Unix systems this is an ISO-8859-x code that is described in ISO standard 8859. For Windows systems, this is a code defined by Microsoft such as, for example, the CP1252 character set with Euro symbol for western Europe.
- BS2000 systems and z/OS computers, on the other hand, normally use an EBCDIC (Extended Binary-Coded Decimal Interchange Code).

Data conversion between openFT partners always applies to the characters with which parameter values (e.g. file names, user IDs, follow-up processing strings, etc.) are transferred.

The conversion of file contents, by contrast, is only relevant for files to be transferred in text format; no data conversion is performed by openFT when transferring files in other formats (binary, transparent, etc.).

Please make sure that the openFT partner codes use the same character repertoire. If this is not the case, some of the characters in the text file (e.g. umlauts) may not be represented correctly. If you transfer files with openFT partners as of V10, you can assign the "Coded Character Sets" that are to be used for local and remote data conversion in the request. It is also possible to transfer Unicode files with these partner systems, see section “Transferring 7-bit, 8-bit and Unicode files”.

2.1.2 openFT product range

The tables below provide an overview of the openFT product range, showing the openFT products currently available for your computer.

openFT product range

Product	Operating system	Comment
openFT (Unix systems)	AIX, Linux, HP-UX, Oracle Solaris [1]	Intel x86 architecture and IBM z processors, additional systems on request
openFT (Windows)	Microsoft Windows	x86 architecture
openFT (BS2000)	BS2000	BS2000 systems from Fujitsu Technology Solutions
openFT (z/OS)	z/OS	z/OS systems from IBM

openFT add-on products

Product/delivery unit	Operating system	Comment
openFT-FTAM (Unix systems)	AIX, Linux, HP-UX, Oracle Solaris [1]	Intel x86 architecture and IBM z processors, additional systems on request
openFT-FTAM (Windows systems)	Microsoft Windows	x86 architecture
openFT-FTAM (BS2000)	BS2000	FTAM functionality for BS2000 systems from Fujitsu Technology Solutions
openFT-FTP (Unix systems)	AIX, Linux, HP-UX, Oracle Solaris [1]	Intel x86 architecture and IBM z processors, additional systems on request
openFT-FTP (Windows systems)	Microsoft Windows	x86 architecture
openFT-FTP (BS2000)	BS2000	FTP functionality for BS2000 systems
openFT-FTP (zOS)	z/OS	FTP functionality for z/OS systems
openFT-AC (BS2000)	BS2000	FTAC functionality for BS2000 systems
openFT-AC (zOS)	z/OS	FTAC functionality for z/OS systems
openFT-CR	All platforms of the openFT product family [2]	Data encryption (restricted to export)

[1] As of openFT V12.1B, the release is only for Linux x86_64. Since openFT version V12.1C20, Solaris (Sparc) is also included in the release again. Further platforms are available on request from your Fujitsu sales representative.

[2] As of version V12.1C openFT-CR is integrated in openFT.

2.2 Heterogeneous networks

A group of interlinked computers and other devices is referred to as a network. When computers with the same type of communication structure are linked, we use the term homogeneous network.

The term heterogeneous network is used to denote a computer network in which computers intercommunicate with different communication architectures. Essential properties of computer networks are distances to be covered, the type transmission route, the utilization of public services and the type of protocols, i.e. the entire range of rules and regulations which must be observed for information transfer.

The most renowned networks supported by openFT are TCP/IP, ISO, SNA, X.21/X.25, ISDN. Not all network types are supported on all platforms.

Network management in heterogeneous networks are based on **SNMP** (Simple Network Management Protocol) in most cases.

The openFT products on BS2000 and Windows support the SNMP-based network management and thus underline their import in open networks.

2.2.1 The OSI reference model

In order to exchange data, systems must be able to intercommunicate. Communication is possible only if the computers involved use the same file formats for data exchange and observe an agreed behavior during transfer. The sum of the conventions and file formats for communication is referred to as a protocol. Protocols are defined by the manufacturer (for example openFT protocols) on the one hand, and on the other by committees which define manufacturer-independent protocols. ISO (International Organization for Standardization) provides the OSI Reference Model (**O**pen **S**ystems **I**nterconnection), the bestknown model for communications architecture and the most comprehensive collection of protocols.

The OSI Reference Model structures the communications functions of computer systems and provides a foundation for standardization of protocols and services. It specifies which functions the components involved in communication must provide.

The OSI Reference Model consists of seven hierarchically structured layers. Each layer is assigned specific communication functions.

Layers	**Designation**	**Functions**	
Layer 7	Application Layer	Coordinates and controls the performance of communication tasks for an application	**A** P P L I C A T I O N
Layer 6	Presentation Layer	Regulates the form of information presentation and thus permits user/device-independent communication	
Layer 5	Session Layer	Regulates the sequence of communication	
Layer 4	Transport Layer	Regulates the reliable exchange of data between two communications partners	**T** R A N S P O R T
Layer 3	Network Layer	Regulates the exchange of data between two terminal systems (computers)	
Layer 2	Data Link Layer	Secures the transmission on individual subroutes of the entire transmission route (procedures)	
Layer 1	Physical Layer	Provides the physical connection (via the medium used for transmission)	

OSI Reference Model

The individual layers use the service of the layer immediately below and provide a precisely defined service to the layer above. Only the physical layer must provide its service together with the physical medium. The active elements within a layer, which provide the functions, are referred to a instances.

Each layer is specified by the service it provides, and the services it uses from the layer below it. During communication, the various computers interoperate on the same layer, using common protocols.

The functionality of each layer in the OSI Reference Model can be provided by various protocols as a rule. Decisive for the communication is that the direct partner instances use the same protocol for a particular task. For this purpose, profiles are defined.

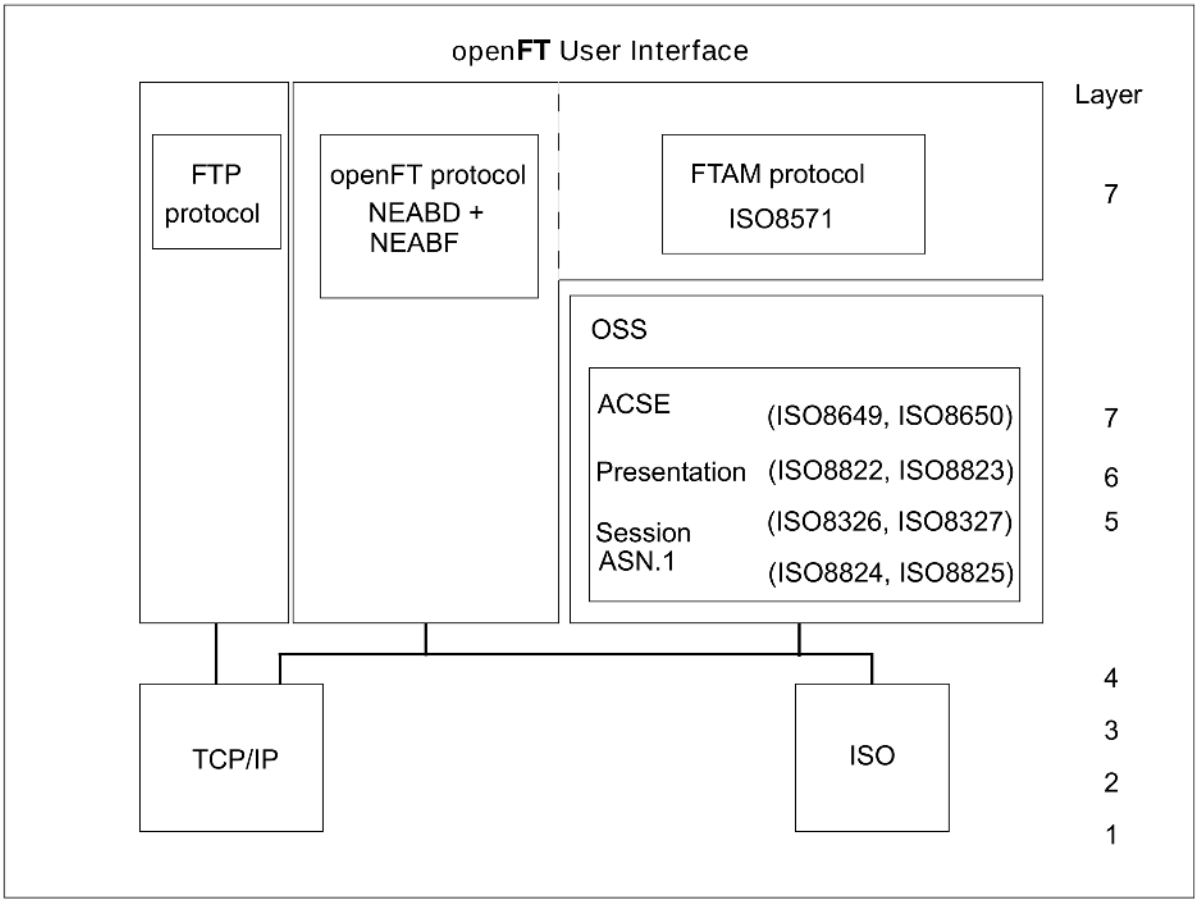
A profile is understood as precise specification of which protocols or which protocol variants are to be used on which layer to perform a particular task. Profiles are stipulated by national or international organizations or communities.

2.2.2 Position of the openFT product family in the OSI Reference Model

The openFT products belong to the application layers (Layers 5 - 7) of the OSI Reference Model. They support the standardized openFT protocol and the FTAM protocol ISO8571 standardized by ISO and the File Transfer Protocol (FTP) defined by RFC959.

The openFT products can use a variety of different transport systems with different transport protocols.

The following diagram shows the possible combinations of application and transport protocols for file transfer:



Protocols supported by openFT in the environment of the OSI Reference Model

openFT for z/OS supports only the openFT protocol and the FTP protocol for transferring files.

For an overview of the transport systems and protocols that permit the operation of openFT products, please refer to the relevant product data sheets.

2.2.3 openFT partners

openFT can perform file transfer and file management between partner systems which support the openFT protocols NEABD and NEABF in the application layers.

These partner systems are referred to below as openFT partners. openFT partners can run on mainframe platforms (BS2000 systems, z/OS) and on open platforms (Unix systems, Windows systems).

Depending on the particular transport system software, a variety of transport protocols may be used:

- TCP/IP transport protocols (all platforms)
- SNA transport protocols (z/OS)
- ISO transport protocols (BS2000 systems, Unix systems and Windows systems)
- X.25 transport protocols (Linux and Windows systems)

The range of functions is largely identical for a given openFT version across the different platforms, and any minor differences are the result of the operating system used.

> ¡ These protocols, which were originally referred to as FTNEA protocols, have been opened, so there are now also products from other manufacturers that support these protocols.

2.2.4 FTAM partners

The FTAM extension available in openFT also enables openFT to perform file transfer and file management with partner systems which support ISO protocols in layers 5 - 7 of the OSI Reference Model. In the rest of this manual, these systems are referred to as FTAM partners, since they use the protocols for file transfer defined in the international standard ISO 8571 (FTAM, File Transfer, Access and Management).

BS2000 systems also require the OSS software package to implement layers 5 - 7.

FTAM is not available on z/OS.

Implementation of FTAM Standards in openFT

A subset of the complete functional scope of the base standards has been selected in accordance with international and European profiles ISO/EN ISP 10607-3 and ISO/EN 10607-6. This functional standardization has, in turn, been harmonized with other functional standards (and implementation agreements), e.g. the corresponding implementation agreements of IGOSS in North America and corresponding profiles in Asia and Australia.

ENV 41204 and ENV 41205 are the old, nevertheless still applicable, designations for EN 10607-3 and EN 10607-6 and their contents are identical to the international profiles ISO/IEC ISP 10607-3 (1990) and ISO/IEC ISP 10607-6 (1990) agreed by ISO. EN 10607-3 and EN 10607-6 contain additional European character repertoires.

These profiles specify the file attributes actually used, for example, and the operations permitted with these attributes, irrespective of the operating system used. A **virtual filestore** is used to permit presentation across several operating systems; here, the contents of the real store are transferred with a representation of the file attributes in accordance with the standard. Conversion of the file attributes to FTAM Standard in the operating system and vice versa is part of the FTAM functionality. There are three groups of file attributes: kernel group, storage group and security group (see "Virtual filestore").

Compliance with the FTAM standard also restricts the functional scope offered by openFT protocols. Transfer of follow-up data to FTAM partners is not possible with the protocol.

The mapping mechanism between the real filestore and the virtual filestore is described in detail on "Virtual filestore"
.

2.2.5 FTP partners

If the FTP protocol is used then only communication via TCP/IP is possible. Furthermore, a number of special considerations apply when FTP servers are used compared to openFT partners. These are for the most part due to limitations in the FTP protocol:

- No restart is performed.
- Encryption is only possible for outbound requests to an FTP server that provides support for Secure FTP with the TLS protocol. This requires openFT-Crypt (openFT-CR delivery unit) to be installed.
- Follow-up processing and coded character sets are only supported locally.
- Several file properties are not transferred. For details, please refer to the corresponding „command interface“ manual.

Please note that the other openFT functions (preprocessing and postprocessing, FTAC, etc.) can only be used if openFT is used as the FTP server on the system, where preprocessing and postprocessing are to be performed.

2.3 Transferring files

The main function of openFT is to transfer files between two partner systems. To do this, you must issue a file transfer request in the local system. This request can be used either to send a file to a remote system or to fetch a file from a remote system to the local system. A partner system can also send files to your local system or fetch one from your local system.

Requests issued from your local system are referred to as **outbound requests** . Requests issued from the remote system are referred to as **inbound requests**.

In a file transfer request, you can specify whether the file to be transferred is a text file or whether it contains unstructured or structured binary data. This determines the handling of the data during transmission; see the section “File conversion”. The so-called “transparent” file format plays a special role here: you can use this format to store BS2000 files with all their properties in the receive system without conversion. This is necessary, for example, when a Unix or Windows system is used to distributed BS2000 software.

On Unix and Windows systems, it is also possible to send a complete directory to a partner system using a single request.

Preprocessing, postprocessing and/or follow-up processing can be agreed for all file transfer requests to openFT partners. You may specify follow-up processing for successful and failed transfers both in the local system and in the remote system. For details of how to use the preprocessing. postprocessing and follow-up processing features, see the section “File transfer with preprocessing, postprocessing and follow-up processing”.

You should not process a file further until transfer is completed; otherwise, inconsistencies may result. On BS2000, DMS files are protected during the entire execution of the request.

You may decide when openFT is to carry out your transfer request. Either immediately or at a particular time which you can specify. openFT always performs a synchronous request immediately. If a request is to be performed later, you must start an asynchronous request and specify the time of its execution.

Compressed transfer

When issuing a request, you may specify whether the file is to be transferred in a compressed form and the type of compression that is to be used (byte compression or zip compression).

Data compression can be used to:

- shorten transmission times
- reduce the load on the transmission paths and
- reduce data transmission costs.

2.3.1 Specifying the transfer start time

When you start a **synchronous request**, the file is transferred immediately. You have the advantage of knowing immediately whether or not the transfer was successful. On Unix and Windows systems, a display on screen allows you to follow the progress of the file transfer during the entire transmission period.

You can use the result as decision criterion for further steps. If transfer failed because the partner was not available, for example, the file transfer is aborted and you can restart the request later.

In the case of an **asynchronous request**, openFT transfers the file either at the next possible time or at the time you specify. This allows the file transfer to be started at a time when the partner is available, or when transmission charges are particularly low. The request is stored in a request queue and you receive confirmation that the request has been accepted. Your system is thus immediately free for other tasks and you do not have to take care of executing the request. Thus, for example, if it is not possible to set up a connection for file transfer at a particular time, openFT re-attempts start of file transfer at defined intervals; even if a fault occurs during transfer, it is restarted automatically. The restart function for asynchronous requests can also be deactivated partner-specifically and via operation parameter settings.

You can start several asynchronous requests. The requests are placed in a request queue until they are successfully executed, or cancelled by you or their maximum lifetime as set globally has been reached (see the section “Controlling the duration of a request”). You can use the request queue to obtain information on all request that have not yet been executed.

Requests issued by a remote system, i.e. inbound requests, are always executed as asynchronous requests in the local system by openFT.

2.3.2 Controlling the duration of a request

An asynchronous openFT request remains in the request queue until it is fully executed or explicitly deleted or until its lifetime, which can be set via an administration parameter, expires.

When issuing an asynchronous request, however, you may specify a time at which the request is to be deleted, or the file transfer is to be canceled (cancel timer). In this way, you can avoid tying up resources for partners who are temporarily unavailable, or when network problems are encountered.

2.3.3 Request queue

The request queue stores all asynchronous file transfer requests which have not yet been executed. You may display these on screen at any time. The information displayed will include:

- the transfer direction
- the operational status of the request
- the number of bytes already transferred
- the initiator of the request
- the local file name, for outbound requests also the remote file name.
- the partner system involved
- follow-up processing
- diagnostic information

The byte counter in the request queue is updated at regular intervals, so that you can keep up-to-date on the progress of file transfer.

You may delete requests and change the order of the requests in the request queue (priority control).

For information on requests that have already been completed, use the logging function (see the section “Logging openFT operations - the logging function”.

Priority control

The requests are processed according to the FIFO principle (FIFO = First In First Out), i.e. the request issued first is processed first. Multiple priority classes are possible. You can control the processing of a request by:

- explicitly specifying the priority of a request
- changing the priority of a request in the request queue
- changing the queue of the request queue, i.e. placing requests at the start or end of a list of request with the same priority

In addition, the FT administrator is able to prioritize partners, see "Prioritization of partners".

2.3.4 Automatic restart

In the event of file transfer being interrupted for any reason, openFT provides for secure restart. This means that network problems, for example, present no difficulty to openFT, since openFT automatically continues transfer as soon as it becomes possible again.

The storage of the request in the request queue and the so-called restart points for the basis for automatic restart. These are the security points with which the two partner systems are synchronized at regular intervals during file transfer. If transfer is interrupted, it is continued as soon as possible starting at the last security point. You can therefore rest assured that not one single bit is lost and nothing is added during file transfer.

The fixed timing between security points ensures that no unnecessary security points are set for fast lines, and that the intervals are not too long for slow lines.

> The automatic restart may not be performed for directory transfer requests if a directory to be transferred is having a sub directory.

Deactivation of automatic restart

It is sensible in some situations to deactivate the restart function for the openFT and FTAM protocol for asynchronous file transfer jobs. If, for example, an openFT Initiator instance is coupled to a load balancer which selects one of the many responders depending on the load, there is no guarantee that this load balancer at restart selects the same responder instance which processed the job at the beginning. The restart would fail and a job "corpse" remains in the original responder.

The partner-specific deactivation of the restart function has an impact on outbound requests. The restart function can also be deactivated via operation parameter settings, separately for outbound and inbound requests.

2.4 File management

In addition to file transfer, openFT offers the option of managing files in local and remote systems. You can perform file-management actions both with openFT statements and as processing within a file transfer request. It can be useful, for example, to set up the necessary conditions for transfer or follow-up processing in the remote system prior to start of file transfer.

Furthermore, local or remote systems can be controlled from a Windows or Unix system via a user-friendly interface similar to the Windows standard, without the user having to be acquainted with the syntax of the remote system.

You can perform the following actions with via file management:

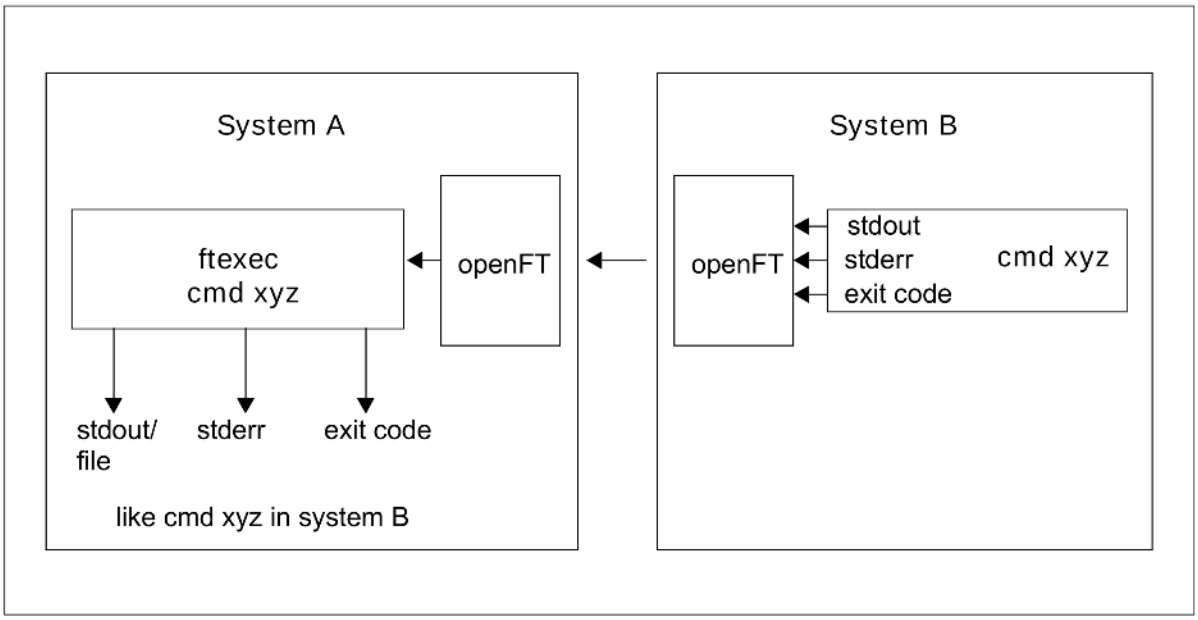
- rename files
- delete files
- query file attributes, e.g. the size of a file
- modify file attributes, e.g. access rights
- display directories
- create directories
- rename directories
- delete directories

2.5 Remote command execution

openFT enables operating system commands to be executed on remote systems (ftexec mechanism) and can return the exit codes and outputs of such commands to stdout and stderr (i.e. SYSLST and SYSOUT in BS2000) as if they were executed on the local system.

This makes it possible to integrate remote commands transparently in local command procedures.

The following diagram clarifies the concept of remote command execution.



openFT concept for remote command execution

2.6 Automation

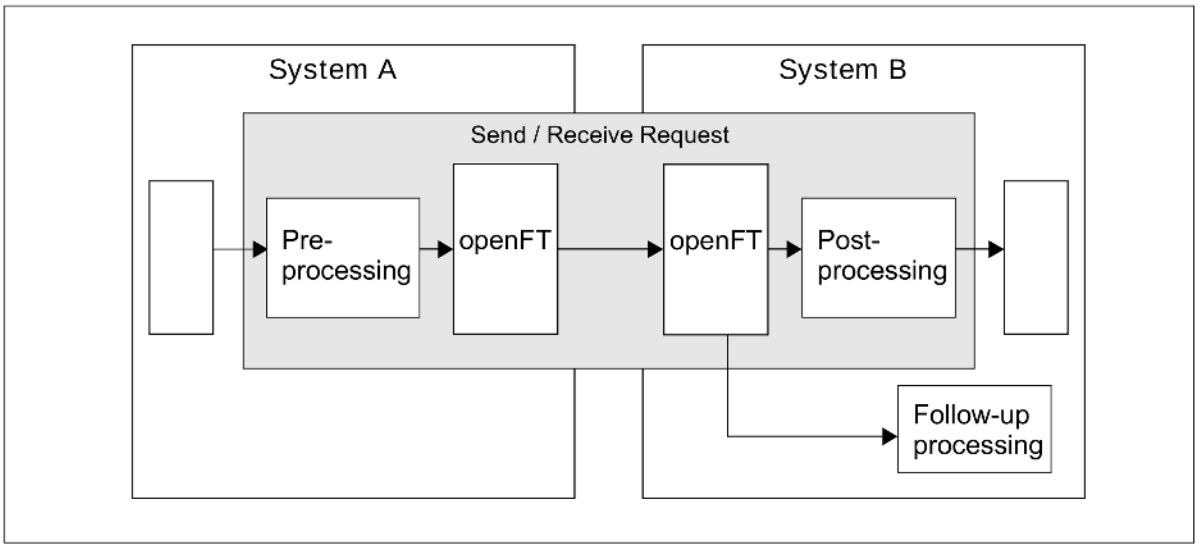
openFT provides job management functions such as file transfer with preprocessing, postprocessing and follow-up processing, the use of Monitor Job Variables in BS2000, and the use of file-transfer functions in dialog procedures and via program interfaces. Automation is also supported by the option for controlling the start time and lifetime of requests; see the corresponding sections. The creation of unique file names by using openFT variables makes it easier to design applications and reduces the amount of updating work to be done.

2.6.1 File transfer with preprocessing, postprocessing and follow-up processing

For a file transfer, you can specify

- whether any preprocessing or postprocessing is to be done within a request. Preprocessing in the sending system and postprocessing in the receiving system are always possible and can also be combined within a request.
- whether any follow-up processing is to be performed after the file transfer. Follow-up processing can be defined for successful and unsuccessful file transfers both for the local and the remote system.

The following diagram clarifies the concept of a file transfer with preprocessing, postprocessing and follow-up processing.



openFT concept for preprocessing, postprocessing and follow-up processing

Pre- and postprocessing always take place within the openFT request, and follow-up processing always take place after the request.

In order to prevent system resources from being unnecessarily tied-up (by deadlocks or pending processes, for example) in a continuous processing loop, requests should be provided with a specified abort time if necessary.

2.6.1.1 Preprocessing and postprocessing

During preprocessing, you can, within a file transfer request, prepare the send data **before** the transfer. These could be operating system commands, program calls or procedure calls, in order to create or prepare the data before the transfer. The commands can, for example, extract information from a large data base (data base query), or prepare data (compress, encrypt), in order to subsequently pass it to openFT for file transfer.

During postprocessing you can, within a file transfer request, process the received data using one or more commands **after** the actual transfer. To do this, you can execute commands, e.g. operating system commands, a program call or a procedure call. The command(s) can, for example, decode/uncompress data which has been encrypted or compressed using external routers.

> openFT requests with remote preprocessing or postprocessing can also be transferred by older versions of openFT or FT. It is important that a version of openFT that supports preprocssing and postprocessing is used in the remote system.

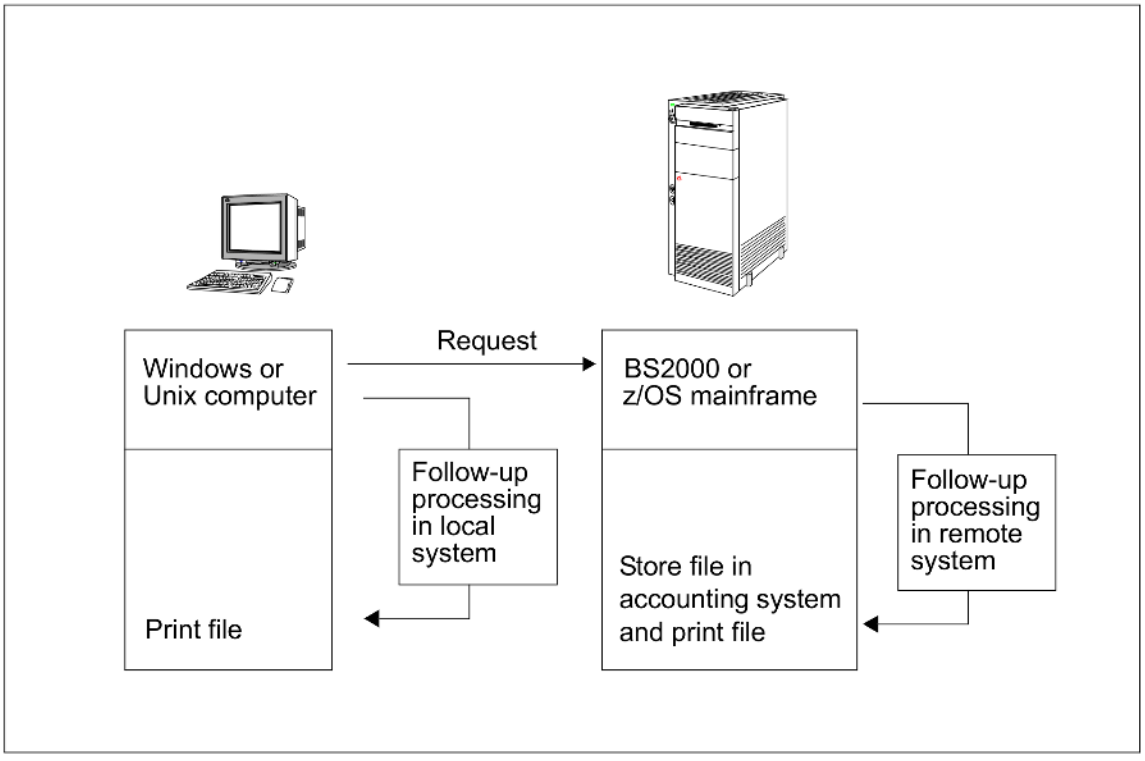
2.6.1.2 Follow-up processing

The "follow-up processing" option which is available in openFT enables you to execute sequences of statements or commands in the local and/or remote system depending on the positive or negative result of file transfer. If you specify follow-up processing for the remote system, you must apply the syntax of the operating system used on the remote system. When using commands, openFT provides variables which are replaced by the values in the file transfer request when the commands are executed.

For directory transfer, the follow-up processing is executed for each individual file transfer request, but not for creating sub directories.

Example

In the headquarters of a supermarket chain, there is a mainframe computer running BS2000 or z/OS. The branch office has Windows or Unix workstations. Every Saturday, the branch manager issues a request to transfer the file that contains a prepared list of the weekly sales. This file is transferred to the processor at the headquarters using openFT. The follow-up processing for the transfer request specifies that the file should be printed on the mainframe and then deleted from the branch computer if file transfer is successful.



File transfer with follow-up processing

2.6.2 Program interfaces

The program interface in openFT offers extensive automation capabilities. You can, for example, automate the issue of requests and request management in openFT, create your own user interfaces for openFT or integrate file transfer functions in other applications. In addition to the Java and C interface, an OCX interface is provided for Windows systems.

2.6.3 openFT script interface on Unix and Windows systems

openFT-Script provides a script language in XML notation which comprises the following openFT functions which are familiar to users from the command or C interface:

- Asynchronous file transfer
- Create directories in the remote system
- Delete files or directories in the remote system
- List directories in the remote system
- Run command scripts in the remote system

All openFT-Script functions can also be applied to local files or directories.

In addition, openFT-Script possesses the following advantages compared to the abovementioned interfaces:

- Logically interdependent individual requests can be combined in a single request thus permitting simple monitoring.
- Individual requests can be run in sequence or in parallel.
- openFT-Script can restart. If an openFT-Script request is interrupted at a specific individual request then the openFT-Script request is resumed at this point on restart.
- openFT-Script requests can be monitored and interrupted in the openFT Explorer via the *Ftscript Requests* object directory.
- Alternative actions can be defined if errors occur (e.g. partner not accessible, file not present etc.).

2.6.4 Job variables in BS2000 systems

openFT (BS2000) offers the option of using a job variable to monitor a file transfer request. The name of the job variable, as well as any password required to access this variable are specified with the file transfer request. The job variable that monitors a request is also used to identify the request and may be used as a selection criterion to abort the request or obtain information about it. Further, it can be used for event control purposes, or to change request priorities.

2.6.5 Automated further processing of openFT data

In order to permit openFT data (e.g. log records) to be automatically processed further by external procedures or programs, openFT offers the so-called CSV (**C**haracter **S**eparated **V**alues) output format. The CSV-format is available for all display commands of openFT. In this format, each block of information is output to one line of text, with the individual items of information in an "output record" being separated by semicolons. The first line is a header and contains the names of the items of information, also separated by semicolons.

Such output could then be processed further by programs which support CSV formats (e.g. Microsoft Excel™ under Windows) and could hence be used, among other things, to easily implement an accounting system for the used resources (e.g. transfer requests).

2.7 Secure operation

Open networks, security during file transfer and data management are terms that need not be contradictory. openFT offers the following functions for secure operation:

- individual settings for transfer and access rights with the FTAC function
- data encryption during the transfer
- logging function that can be enabled/disabled
- Checking the communication partner using authentication

You can use these functions to make your system safe.

2.7.1 The FTAC function

With the FTAC function of openFT, you have all the options in your hand to make your system as secure as possible and as safe as it needs to be. FTAC stands for “File Transfer Access Control”.

FTAC offers the following protection mechanisms for your system:

- decoupling of FT transfer and login authorizations
- access rights dependent on the partner systems
- user-specific access rights
- flexible access right levels
- recording of every authorization check
- simple application

2.7.1.1 Features of the FTAC function

For file transfer, a distinction is made between various functions. For access protection, the file transfer function being executed by the system is decisive. At first glance, there are only two such functions:

- sending a file and
- receiving a file.

Sending a file entails transmitting data from the system to be protected, while receiving a file involves the transfer of data into this system. However, for reasons of data security it is also important to know who requested a function in the system being protected. In FT terminology, this person is referred to as the initiator or submitter of the FT request. Initiators can be divided into two groups:

- those in the system being protected (**outbound requests**)
- those in partner systems (**inbound requests**)

With this information, we can now make a distinction between four basic functions:

- **Outbound send**
- **Outbound receive**
- **Inbound send**
- **Inbound receive**

The possibility of processing transfer data (pre-, post-, and follow-up processing) during a file transfer should be considered an additional function. For FT requests submitted in the local system, no additional protection is necessary since anyone in the local system allowed to initiate FT requests already has access to the available resources. Processing in the remote system does not require any protective measures in the local system either. One function that does require protection in the local system is

- **Inbound processing**

which is initiated from a remote system.

Partner systems also have the option of using the file management functions to view directory or file attributes in their local system, to modify file attributes and to delete files and directories. This results in a further function:

- **Inbound file management**

File management, unlike the other functions, encompasses several different request options, which in turn are partially linked to the functions *inbound send* and *inbound receive*:

Inbound file management function	**Prerequisite**
Show file attributes	Inbound send permitted
Modify file attributes	Inbound receive **and** inbound file management permitted
Rename files	Inbound receive **and** inbound file management permitted

Delete files	Inbound receive permitted

The protection mechanisms offered by the FTAC function are primarily achieved through the use of admission sets and admission profiles.

2.7.1.2 Admission set

When FTAC is used, an individually configurable admission set is assigned to every user ID in the system. There is also the so-called **standard admission set**, which is defined by the FTAC administrator and contains the preliminary settings and defaults for all the individual admission sets.

The admission set contains the basic specification of the file transfer functions. It is possible to define security levels between 0 and 100 for each of the six file transfer functions. 0 means that the function is blocked and 100 that the function is available without restriction. When access is attempted under a user ID, FTAC checks whether the security levels set in the corresponding admission set are complied with.

2.7.1.3 Admission profile (FT profile)

The admission profile (or FT profile) defines the **FTAC transfer admission** and the associated **access rights**. The FTAC transfer admission (simply referred as transfer admission) is the actual key to your computer. You should therefore treat the transfer admission with the same care as you look after a password. It must be specified in transfer requests instead of a login authorization. The standard admission profile for a user ID is an exception. See "Admission profile (FT profile)". Anyone who possesses this transfer admission does have file transfer access to your computer, but, unlike the login authorization, is not free do what he or she wants. Which functions you permit are specified with the access rights for this transfer admission. In this way, you can control (for example):

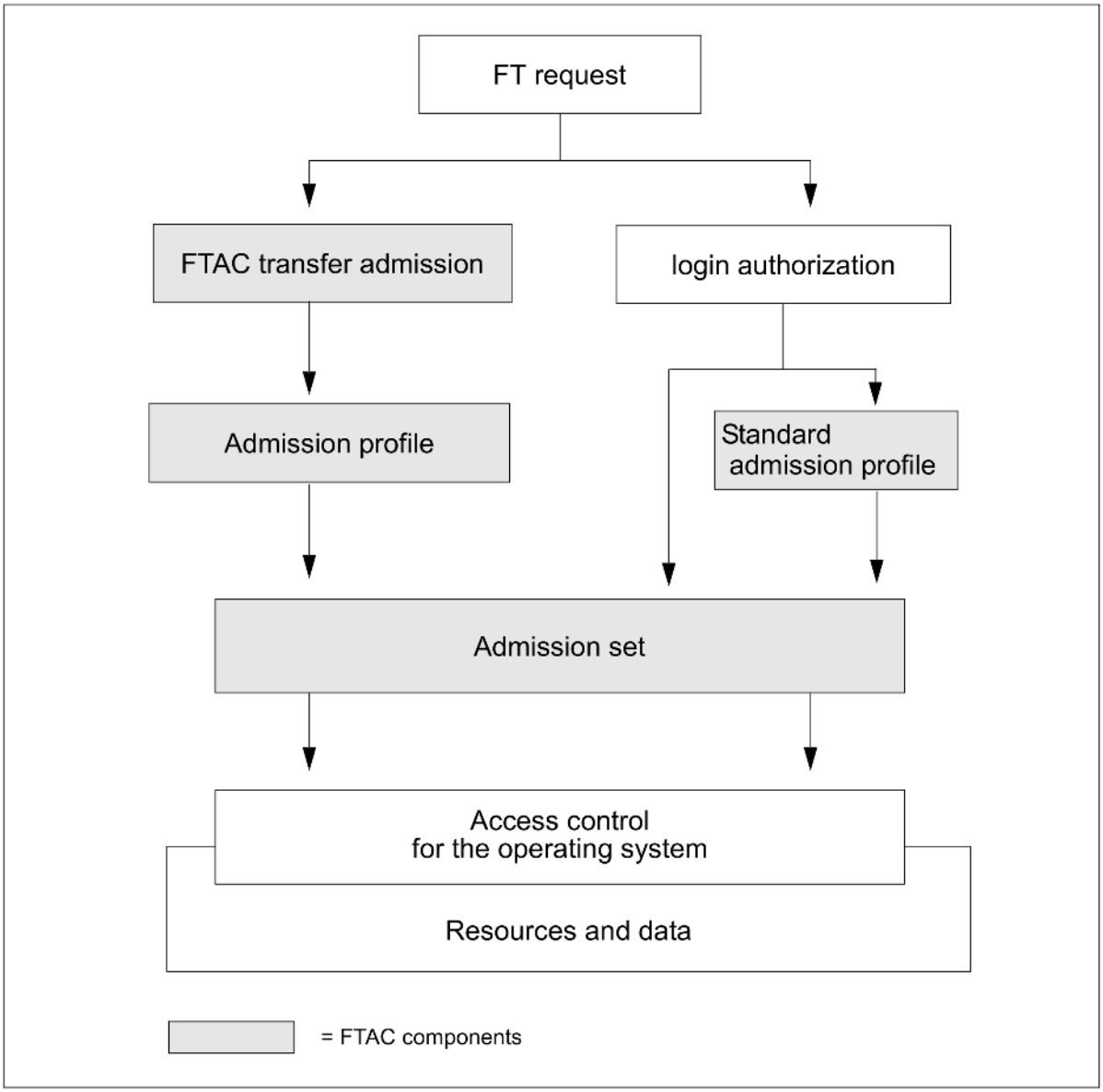
- the conditions under which files are accessed (file name prefix)
- the follow-up processing commands which are permitted after file transfer
- the partner systems which can access the admission profile.

In the most extreme case, you can restrict access to your computer in such a way that only one single profile is available providing access to only one file.

FTAC checks whether the entries in the request conflict with the entries in the FT profile for each file transfer request. If so, the file transfer request is rejected. In this case, only a general error message appears in the remote system.
This prevents the definition of the FT profile being determined step-by-step on a trial and error basis. A log record which describes the cause of the error precisely is created in the local system.

Admission checking workflow

The following diagram shows the sequences for admission checking with FTAC.



Access check with FTAC

Standard admission profile

You can set up a standard admission profile for each user ID. Unlike a normal profile, a standard admission profile has no FTAC transfer admission.

This profile is only intended for certain use scenarios, such as when an FTAM partner has to specify the transfer admission in a fixed structure (user ID and password) for inbound access and you nevertheless wish to specify certain settings, such as a filename prefix.

2.7.2 Encryption for file transfer requests and file management requests

openFT offers the following encryption possibilities for data transfer and file management:

- Encryption of request description data:
  This refers to the protocol data which file transfer sends and receives in order to establish connections and process requests.
- Encryption of user data (i.e. the content of the transferred file, only possible for openFT-Partners and outbound secure FTP partners):
  The encryption of the file contents can be set individually for the transfer request or via an admission profile or, more generally, in the operating parameters. It is also possible to force or prohibit encryption, for example for performance-related reasons.
- Encryption of file attributes and directory list attributes

When connecting to partners that support the AES algorithm, then RSA/AES encryption algorithm is used for the request description data and the content of the transferred file.

To do this, openFT as of V12.0 uses a 256-bit AES key and a 2048-bit RSA key by default. Alternatively, a 1024-bit or 768-bit RSA key can be used, for Unix and Windows systems additional a 3072-bit or 4096-bit RSA key. The FT administrator must set this in the operating parameters. In the case of connections with older versions, encryption is negotiated downwards if necessary, i.e. an RSA of a length that is available in the older version is used or, if AES keys are not supported, DES encryption is employed.

The FT administrator can also set a minimum RSA key length and/or a minimum AES key length. In this case, a request is rejected if the partner cannot fulfill this requirement.

The mechanism for active encryption of user data is a separate supply unit and must be unlocked explicitly due to legal requirements.

For further details, see section “Encryption for file transfer and file management” and section “Protection mechanisms against data manipulation”.

2.7.3 Logging openFT operations - the logging function

Prevention of unauthorized access and protection of data inventories is just one security aspect. The complete documentation of the access check and the file transfer requests also puts you in a position to check your security network at any time and detect any leak. The logging function of openFT is the most suitable tool for doing this. It is activated as default and logs all information relating to file transfer requests, irrespective of whether the initiative lies in the local or remote system and whether the transfer was successful or not. The **log records** are written into the corresponding file. The scope of logging can be set as appropriate.

The logging function also serves as a basis for detecting break-in attempts. In addition, it may be used to obtain and evaluate performance data (see also the section “Automated further processing of openFT data”).

Log records

If your local system is protected by FTAC, FTAC first checks all accesses to your system and logs the result in an **FTAC log record**. If the access check is negative, FTAC already rejects the request. If the access check is positive, the following applies:

- In the case of a file transfer request (and if the request materializes), an **FT log record** is subsequently written indicating whether the request was executed successfully or why it was cancelled. This means that there can be two log records for one transfer request.
- In the case of a remote administration request, an **ADM log record** is written indicating whether the request was executed successfully or why it was cancelled.

You may display log records relating to your login name at any time. Log records of other login names can only be viewed by the administrator.

The administrator can define the scope of logging separately for FT, FTAC and ADM logging and for directory transfer logging. FT, ADM and directory transfer logging can be fully disabled. Although FTAC logging can also be restricted, rejected FTAC access checks are always logged.

Offline logging

The FT administrator can switch the log file during system operation. Following the switchover, new log records are written to a new log file. The previous log file remains available as an offline log file. You can continue to view the log records for your user ID using the tools available in openFT.

Logging request with preprocessing / postprocessing

For security reasons, only the first 32 characters (or 42 characters for *ftexecsv* preprocessing on Unix systems or Windows systems) of a preprocessing or postprocessing command are recorded in the log record.

Saving and deleting log records

Only the FT administrator, the FTAC administrator and the ADM administrator are permitted to delete a log record or log file. Log records should be saved at regular intervals (ideally using a cyclical job). During this, the output of the corresponding command, not the active log file itself, should be saved. Switching the log file makes it possible to save the current log records in an offline log file. This offline log file can then be backed up by the FT administrator.

The benefit of this is, first, that the log records provide a complete record of FT operations which can be maintained for long periods, and second, that the log file does not assume unnecessarily large proportions, which saves CPU time when accessing the records.

2.7.4 Authentication

If data requiring an extremely high degree of security is to be transferred, it is important to subject the respective partner system to a reliable identity check (“authentication”) before the transfer. The two openFT instances engaged in the transfer can perform mutual checks on one another, using cryptographic resources to determine whether they are connected to the “correct” partner instance.

To this end, openFT supports the following addressing and authentication concept:

- the addressing of openFT instances via network-wide, unique instance IDs:

  For the local system, these IDs are defined using an operational parameter. Instance IDs of partner systems are stored in the partner list. openFT administers the resources assigned to these partners, such as request waiting queues and cryptographic keys, with the aid of the instance IDs of the partner systems.
- the exchange of partner-specific key information:

  The FT administrator can prepare RSA key pair sets, each of which consists of a private and a public key, for each local openFT instance. In order that one’s own openFT instance can be authenticated in the partner system, the appropriate public key must be made available to the partner system. This should take place via a secure path.

For more details, please refer to section “Authentication”.

2.8 Using openFT in a cluster or HIPLEX/Sysplex composite

In openFT you can simultaneously execute more than one openFT instance on a single host. This allows you to switch to the openFT functionality on a different computer in the cluster that is already running openFT when your computer fails.

openFT commands that can be called during preprocessing, postprocessing or follow-up processing execute in the same instance as the request that initiated the preprocessing, postprocessing or follow-up processing.

3 Partner concept

The remote system is referred to as the partner system or simply as the "partner". The openFT partner concept is based on the so-called partner list.

In addition, you can also use TNS on Unix and Windows systems on which PCMX is installed, see section “Using PCMX and TNS in Unix and Windows systems”.

3.1 Partner list

The partner list is set up and maintained by the FT administrator.

A partner in the partner list is defined by its address, see section Partner addresses. In addition, it is possible to assign the following properties:

- Partner name
- Priority
- Instance ID of the partner (if this differs from the default)
- Security levels such as FTAC security levels, see "FTAC security levels for partner entries"
- Serialization settings, see "Serialization of asynchronous outbound requests"
- Inbound/outbound deactivation, see "Outbound and inbound deactivation of named partners"
- Trace options (On/Off)

Depending on whether and in what form a partner system is entered in the partner list, a distinction is made between named partners, registered dynamic partners and free dynamic partners, see section “Partner types”.

3.1.1 Partner addresses

A partner address in openFT has the following structure:

[protocol://]host[:[port number].[T Selector].[Session Selector].[Presentation Selector]]

host (= computer name)

This specification is mandatory; all other specifications are optional.

protocol://

Protocol stack via which the partner is addressed.

Possible values for *protocol* are:

- **openFT** (openFT protocol, default value)
- **ftam** (FTAM protocol)
- **ftp** (FTP protocol)
- **ftadm** (FTADM protocol for remote administration requests).

port number, *T Selector, Session Selector*, *Presentation Selector*

These are only required in special cases or for FTAM partners.

For further details, refer to the openFT command manuals.

3.1.2 Partner types

The partner list plays an important role during the administration of partners. A distinction is made between three types of partner systems depending on whether and in what form partner systems are entered in the partner list.

- Named partners:
  All partners that are entered with their addresses and names in the partner list
- Registered dynamic partners:
  All partners that are entered with their addresses but without a name in the partner list
- Free dynamic partners:
  All partners that are not entered in the partner list

Registered dynamic partners and free dynamic partners are both simply referred to as dynamic partners.

Named partners

In FT requests, named partners are addressed using the names defined for them in the partner list.

These partners remain in the partner list until they are deleted from it by the FT administrator. If authentication is required for the connection to a partner then this partner should be entered in the partner list.

The use of named partners has the following advantages:

- Complex partner addresses do not have to be specified explicitly in openFT requests.
- Security is enhanced because only partners that are genuinely recognized can be permitted.
- openFT requests are independent of address changes: Procedures, scripts or programs with automated openFT requests use the partner name and do not have to be modified in the case of address changes.

> ℹ Although a named partner can also be connected to via its address, in all openFT tasks such as logging or request queue activities, the partner name is displayed.

Registered dynamic partners

All partners that are entered only with their addresses but without names in the partner list are registered dynamic partners. They can only be accessed via the address and possess at least one attribute that differs from the default value for a free dynamic partner (see section “Free dynamic partners”).

If the FT administrator resets all the attributes for a partner of this type to the default values then this partner is removed from the partner list and becomes a free dynamic partner.

Free dynamic partners

Free dynamic partners are all the partners that are not entered in the partner list. They are therefore not displayed when you enter *ftshwptn* without specifying a partner name or partner address.

Partners of this type can only be connected to via their address and, with the exception of the security level (see section “FTAC security levels for partner entries”), possess default attributes in the same way as when a partner is added to the partner list.

The advantage of the free dynamic partner concept is that users can address any required partners that are not entered in the partner list. This reduces the administrator's workload in terms of administration requirements. The disadvantage lies in the increased security risk and is the reason why you are also able to prohibit the use of dynamic partners.

You can transform a free dynamic partner into a registered dynamic partner as follows:

- Enter a partner address that does not refer to any existing partner list entry
- Define one or more attributes with values other than the default (see above).

> ℹ If the state of a free dynamic partner changes (e.g. to NOCON = Partner not available) and is therefore different from the default value then it is displayed in the partner list. However, it becomes a free dynamic partner again as soon as it once more becomes accessible (ACTIVE status).

Activating/deactivating dynamic partners

As FT administrator, you may prohibit the use of dynamic partners for security reasons by means of operating parameters. In this case, it is necessary to address partners via their names in the partner list. They cannot be addressed directly via their address. Inbound access is then also only permitted to partners that are entered with a partner name in the partner list.

3.2 FTAC security levels for partner entries

If the FTAC functionality is to be used, the FT administrator should coordinate with the FTAC administrator to additionally define the security level relevant to FTAC for each partner in the partner list.

The security levels regulate the degree of protection with respect to the partner system. A high security level is used when a high degree of security is required, and a low level for a low degree of security. When FTAC is first used, the security levels should be assigned in multiples of ten. This leaves the option open to incorporate new partner systems flexibly into the existing hierarchy.

You can either assign security levels explicitly (minimum value 0, maximum value 100) or activate the "Partner attributes" automatic mechanism, i.e.:

- Partners that are authenticated by openFT are assigned security level 10.
- Partners that are known in the transport system are assigned security level 90.
- Partners which are only accessed via their IP address (e.g. FTP partners) are assigned security level 100.

This automatic mechanism can be activated on a partner-specific Basis or globally by means of operating parameters.

> ¡ This automatic mechanism also applies to all partners that are not entered in the partner list (free dynamic partners) irrespective of the settings made in the operating parameters..

If no security level was specified when a partner was generated, then openFT uses the global settings in the operating parameters. Here, it is also possible to specify a fixed security level as the default.

Effect of the security level

The security level of a partner entry is taken into account when a user wants to process a request via this partner entry. FTAC compares the security level of the partner entry with the security level for this function (e.g. inbound sending) specified in the user's admission set. If the security level in the admission set is lower than that in the partner entry, the request is rejected by FTAC. If a privileged FTAC profile is used for the request, the user can override the restrictions defined in the admission set.

3.3 Outbound and inbound deactivation of named partners

You are able to deactivate specific named partners for asynchronous outbound requests or for inbound requests.

In the case of outbound requests, you can also enable automatic deactivation which deactivates the partner for outbound requests after five unsuccessful attempts to establish a connection. Before any further attempt to establish a connection is possible, this partner system must be activated again manually. This prevents costs from being incurred unnecessarily since, under certain circumstances, even unsuccessful attempts to establish a connection may be charged for.

3.4 Serialization of asynchronous outbound requests

You can force the serialization of asynchronous outbound requests for a partner system.

This prevents the "overtaking" effects that can arise when requests are processed in parallel. The following points apply to serial processing:

- A follow-up request is not started until the preceding request has terminated.
- Serialization includes preprocessing and postprocessing operations but not follow-up processing operations because these are independent of the request.

This function can be used, for example, in a branch-head office configuration in which the branches send multiple files to the head office at the same time (daily, weekly or monthly figures). If serialization is enabled for the partner "head office" in the branch computers then each branch computer can have only one active connection to the head office computer at any one time. This prevents bottlenecks at the head office computer of the sort that occur, for example, if the connection limit is regularly exceeded (see also the CONN-LIM operating parameter).

3.5 Prioritization of partners

Partners can be prioritized in the partner list. This priority only applies to requests that have the same request priority, but are sent to partners with different partner priorities. Otherwise, the request priority overrides the partner priority.

The following lists show, for each platform in question, the sequence in which requests are processed if requests with different request and partner priorities are present.

BS2000 and z/OS:

Processing sequence	Request priority	Partner priority
1	high	high
2	high	normal
3	high	low
4	normal	high
5	normal	normal
6	normal	low
7	low	high
8	low	normal
9	low	low

Unix and Windows systems:

Processing sequence	Request priority	Partner priority
1	normal	high
2	normal	normal
3	normal	low
4	low	high
5	low	normal
6	low	low

3.6 Using PCMX and TNS in Unix and Windows systems

eAs of openFT V10, it is no longer necessary to use the Transport Name Service (TNS) for connections via TCP/IP. If, however, you do use TNS, for example because you establish connections using transport systems other than TCP/IP or want to make use of existing TNS entries, PCMX must be installed and operation with PCMX and TNS must have been explicitly enabled in the operating parameters.

The TNS identifies a transport system application (TS application) by a symbolic name known as the GLOBAL NAME. Address information is assigned to this symbolic name. In the case of an FT request, the partner is therefore addressed via the GLOBAL NAME.

Please note that the additional functions of the partner list, such as FTAC security levels or inbound/outbound activation for partners cannot be used.

> **Important**!
> On Unix systems, PCMX is supported and provided with openFT V12.1B10 for the last time. Please switch all TNS entries concerning openFT partners to partner list entries. To do this, the procedure *tns2ptn* is available on Unix and Windows systems.

4 File transfer and file management

File transfer with openFT is initiated by a file transfer request. In the file transfer request, you make entries to specify the partner system, the transfer direction, the file name and file properties. Given the variety of hardware and software platforms supported, the values specified are subject to various different conventions applicable to the operating systems involved in file transfer. Which files can be transferred between two computers depends on whether the file transfer partners are running identical operating systems (homogeneous link), or different operating systems (heterogeneous link). If a partner using the FTAM functionality is involved in file transfer, the link is a heterogeneous one as a rule. The file management offered by openFT allows you to delete, rename files, or change file attributes before or after file transmission or even without file transfer.

The use of the FTAC functionality offers you not only security benefits, but also allows you to make your file transfer operating system independent (see the section “Features of the FTAC function”), provided the appropriate FTAC settings exist on the processors involved in the file transfer.

Entries for file transfer requests

The following sections give you an overview of the entries you have to make for a file transfer request. They are divided into a local, a remote and an optional part. In the local part, you specify the local file name, if necessary, with the directory name and the file passwords. In the remote part, you define the remote file name, the partner computer and the access to this processor (login name and, if necessary, the account number and password or transfer admission). In the optional part, you have the option of specifying transfer modalities, such as file types, and follow-up processing requests, for example.

4.1 File names

The description below provides an overview of the system-specific conventions for entering file names, regardless of whether a local or remote file name is involved. By using the FTAC functionality with an appropriate definition in the admission profile, you can avoid having to enter all or part of the file name (see the section “Admission profile (FT profile)”). In other words, the parts of the file name defined in the admission profile need not specified in the file transfer request again.

4.1.1 Unique file names for receive files

One of the important applications of openFT products is to transfer a file to a target system with automatic follow-up processing of the received file. In many cases, the receive file is actually only an intermediate product of the processing involved. In order to prevent potential conflicts with concurrently running requests in such cases, the metacharacter string %UNIQUE or %unique can be specified in the receive file name as to instruct openFT to create a unique file name. openFT replaces %UNIQUE or %unique with a string which varies with each new call. The string length depends on the platform.

Note that in log records, result logs, the status output of transfer requests, and in messages, the file name is displayed using the values that have already been set for %UNIQUE.

4.1.2 BS2000 file names

Format for BS2000 (DMS)	Meaning
:cat:$user.filename	cat Optional specification of catalog ID; Available characters restricted to A...Z and 0...9; max. 4 characters; must be enclosed in colons; Preset is the catalog ID assigned to the login name in the entry in the user catalog.
	user Optional specification of login name; Available characters A...Z, 0...9, $, #, @; max. 8 characters; must not start with a digit; $ and the dot must be entered; Preset is the catalog login name under which the file is accessed.
	filename File name can be split up into several subnames: $name_1$[.$name_2$[...]] $name_i$ contains no blanks and must start or end with a hyphen; Character set is A...Z, 0...9, $, #, @. File name can be up to 41 characters long, must not start with $ and must contain at least one character in the range A...Z.
:cat:$user.group (gen-no)	cat see above user see above group Name of a file generation group For character set see filename, brackets must be specified max. length 41 characters. (gen-no) (*abs) absolute generation number (1..9999); * and brackets must be specified. (+/-rel) relative generation number (0..99); Signs and brackets must be specified.

:cat:$user. lib/typ /element	cat see above user. see above lib Library name; the rules for BS2000 DMS file names apply.
	typ Element type; Alphanumeric name, 1 - 8 characters in length.
	element Element name; The rules for LMS element names apply; element can be up to 64 characters in length, must not begin with $, and must include at least one character from A...Z.

If files are transferred using POSIX commands, local DMS names must be specified including the catid. Names specified without the catid are interpreted as POSIX files. The prefix "::" can be used for the default catid.

POSIX file names

In the local and /or remote BS2000 operands for the POSIX file names, the POSIX file name must be specified as a C string (graphic string) (i.e. enclosed in quotation marks). This is necessary in order to distinguish between uppercase and lowercase in POSIX file names.

Format for BS2000 (POSIX)	**Meaning**
posix-filename	Character string up to 255 characters long. Comprises either one or two dots, or alphanumeric characters and special characters; special characters must be canceled with \. The character / is not permitted. Must be enclosed in quotation marks if alternative data types are permissible, separators are used or the first character is ? or !. posix-filename must be prefixed with a posix-pathname.
posix-pathname	Input format: [./]$part_1$/.../$part_n$ where $part_n$ is a posix-filename; up to 1023 characters; must be enclosed in quotation marks if alternative data types are permissible, separators are used or the first character is ? or !. posix-pathname must begin with / or ./, or consist of at least / or ./.

4.1.3 File names in Unix systems

Up to 512 characters, where a distinction is made between uppercase and lowercase. It is recommended that the following characters be avoided in file names:
- ? @ # $ ^ & * ( ) ‘ [ ] \ | ; " < >.

4.1.4 Windows file names

File name here refers to the complete pathname.

Up to 256 characters. The following characters must not be used:
| * ? " < >.

No network drives can be specified for remote file names, either when fetching or sending files. Instead, you can specify UNC names.

UNC names

UNC names (**U**niversal **N**aming **C**onvention) are addresses of shared resources in a computer network. They have the following format:

```
\\hostname\sharename\path\file
```

Either the host name or the IP address, for example, can be specified for *hostname*:

```
\\host1\dispatch\catalogs\winterissue.pdf
```

or

```
\\172.30.88.14\dispatch\catalogs\winterissue.pdf
```

4.1.5 z/OS file names

Format (z/OS)	Meaning
':S:first-qual>. filename' or :S:filename	Specification for PS dataset :S: prefix for identifying a PS data set (no restrictions) first-qual “first level qualifier” Specification of login name; Available characters: A...Z, 0...9, $, #, @; max. 7 characters; must not start with a digit or alias name (max. 8 characters) filename partially qualified file name can be split up into several subnames using dots: $name_1$[.$name_2$[...]] $name_i$ is up to 8 characters long; available characters: A...Z, 0...9, $, #, @; must not start with a digit The partially qualified file name can be up to 36 characters long Fully qualified name The fully qualified file name (first-qual.filename) can be up to 44 characters long.
':S:first-qual. gengroup. Gmmmm.Vnn' or :S:gen-group. Gmmmm.Vnn	Specification for absolute file generation :S: prefix for identifying a PS data set (no restrictions) first-qual See “Specification for PS dataset” for syntax gen-group See filename in “Specification for PS dataset” for syntax Exception: partially qualified file name, up to 27 characters; fully qualified file name up to 35 characters Gmmmm.Vnn absolute file generation mmmm absolute generation number (0000 - 9999) nn version number (00 - 99)

':S:first-qual.gen-group(rel-gen-no)' or :S:gen-group (rel-gen-no)	Specification for relative file generation :S: prefix for identifying a PS data set (no restrictions) first-qual See “Specification for PS dataset” for syntax gen-group See gen-group in “Specification for absolute file generation” for syntax rel-gen-no relative generation number 0 = current generation +/-m = 1 - 99 for partially qualified specification (without first-qual and quotation marks) 1 - 255 for fully qualified specification (with first-qual and quotation marks)
':prefix':first-qual. filename (membername)' or :prefix:filename (membername)	Specification for PO or PDSE member :prefix: prefix for identifying the file organization (no restrictions); can have the following values: :O: for PO :E: for PDSE :L: for PO or PDSE first-qual Syntax see “Specification for PS dataset” filename Partially qualified file name of PO or PDSE dataset Syntax see filename in “Specification for PS dataset” membername Name of PO or PDSE member max. 8 characters long, available characters: A...Z, 0...9, $, #, @; must not start with a digit

“:V:first-qual. filename” or :V:filename	Specification for VSAM file of type "entry-sequenced" :V: Optional prefix for designation of a VSAM file of "entry-sequenced" first-qual Syntax see "Specification for PS data set" filename Partially-qualified file name of VSAM file Syntax see filename in "Specification for PS data set"
':prefix: first-qual. filename' or :prefix:filename	Specification for a complete PO or PDSE data set :prefix: prefix for identifying the file organization (no restrictions); can have the following values: :O: for PO :E: for PDSE :L: for PO or PDSE first-qual See "Specification for PS data set" for syntax filename partially-qualified file name of PO or PDSE data set See filename in "Specification for PS data set" for syntax Exception: maximum length of partially-qualified file name is 34 characters, fully-qualified file name is 42 characters. Thus the maximum permitted file name length is, for both partly and fully qualified specifications, 2 characters shorter than for a PS data set. This is because the name of a temporary data set required to transfer a complete PO or PDSE data set is formed by adding ".U", see manual "openFT (z/OS) - Command Interface".

openEdition files

The file names comply with the POSIX conventions.

Format with z/OS	Meaning
filename	Components of an openEdition filename. String up to 255 characters in length. This comprises either one or two periods or alphanumeric characters and special characters. The character / is not permitted.

pathname	openEdition file name Input format: [.]/[$part_1$/.../$part_n$] where $part_n$ is a POSIX file name; up to 512 characters. If the name starts with /, it is interpreted as an absolute path name. If the name starts with ./, it is a “relative” path name and is relative to the directory for the user ID, e. g. /u/userid in lowercase characters/.

4.2 File passwords

If a password applies to a file that is accessed with openFT is password-protected, the password must be entered. In Windows and Unix systems, there are no file passwords.

System	File password
BS2000	1 - 4 character C string (graphic string) or 1 - 8 character X string (octet string) or integer string between -2147483648 and 2147483647
z/OS	1 - 8 alphanumeric characters

4.3 File types

Depending on their file type and the operating system from which they originate, files that can be transferred have different properties, which must be considered during the transfer.

4.3.1 BS2000 files

In accordance with the different file structures, a distinction is made between the following BS000 file types:

- Cataloged files
  - DMS files (these include SAM, ISAM, and PAM files, PLAM libraries and cataloged generations of a file generation group)
  - POSIX files
- Elements of a cataloged PLAM library
  - Printable or user-definable elements of type D, J, M, S and possibly X
  - Elements with BS2000-specific binary code of type C, L, R and possibly X

In order to be able to transfer POSIX files using openFT, POSIX must be started. The POSIX file system essentially corresponds to the layout and structure of the Unix file system.

The following overview shows the relationship between file name syntax and file type in BS2000.

File name syntax	**File type**
Starts with $userid or :catid:$userid and does not contain '/'	DMS file, fully qualified
Starts neither with '/' nor with './' nor with $userid nor with :catid:$userid and does not contain '/'	DMS file, path relative to transfer admission
Starts with '/'	POSIX file, fully qualified
Starts with './'	POSIX file, path relative to transfer admission
Starts with $userid or :catid:$userid and contains at least one '/'	Name of a PLAM element, fully qualified
Starts neither with '/' nor with './' nor with $userid nor with :catid:$userid but contains at least one '/'	Name of a PLAM element, path relative to transfer admission

BS2000 files may be located either on common disks or on private disks. For processing of files on private disks, the files must be cataloged and the private disks must be properly connected to the system.

Transferring library members

FT systems transfer precisely one library member with an FT request. In the case of transfers between library members, the library organization, the access method and the record structure are retained, whereas in the event of transfers between library members and SAM files only the record structure is preserved.

- During file transfer with FTAM partners, library elements that are located in the remote system can only be accessed using a C-string (and not by using *LIB-ELEM()).
- Library members of a specific type (e.g. load modules) cannot be transferred unless the FT product in use at the partner system permits the transfer of members of this type.

4.3.2 z/OS files

openFT (z/OS) can transfer the following types of files:

- PS datasets including absolute and relative file generations
- Members of PO and PDSE datasets (with the exception of object modules and programs)
- VSAM files of type “entry-sequenced”
- openEdition files (files belonging to the z/OS Unix Systems Services)
- Migrated files, i.e. files swapped out with HSM. See also the section “Migrated files”.

The transfer of these files is performed sequentially. The files can be transferred homogeneously between two z/OS systems or heterogeneously with a non-z/OS system. For homogeneous file transfer, all file types can be mapped to one another. Between z/OS and other platforms (heterogeneous link) it is possible to transfer files if the remote system also supports sequential files. With BS2000 systems, for example, SAM files and PLAM elements of the appropriate type can be exchanged.

The transfer of complete PO and PDSE datasets can only take place between two z/OS systems.

z/OS files may be located either on common disks or on private disks. For processing of files on private disks, the files must be cataloged and private disks must be properly connected to the system. For the processing of files on private media, the precondition is that the files are cataloged and that the private data medium has been properly connected to the system.

Files with the attribute “unmovable” (data organization PSU) cannot be transferred by openFT.

4.3.3 Unix and Windows files

Files in Unix systems and Windows systems, like POSIX files in BS2000 systems, have no structure and no file attributes that provide information on the coding. Although they have no structure either, Windows files can be distinguished on the basis of their file extensions (e.g. “txt” for text and “exe” for executable files).

For transfer with Windows or Unix systems, you can therefore define the following file types:

- text
- unstructured binary data
- binary data structured in records (user format)

Text format

A file that is sent in text format from Windows or Unix systems, must be a pure text file with a record structure defined by linefeed characters in Unix systems or Carriage Return and linefeed in Windows. The length of a line is limited, e.g. 98403 bytes in Windows systems. For the transfer, the end-of-line character is removed from every line.

During transfers from BS2000 or z/OS systems to Windows or Unix systems, the end-ofline character is inserted into the sentence length already in the remote system. The text and the sentence lengths are preserved. The line length is restricted, e.g. to 98304 bytes in Windows systems. The maximum sentence length during a text file transfer depends on the operating system.

When communicating with partner systems as of openFT V10, it is also possible to transfer Unicode files; see section “Transferring 7-bit, 8-bit and Unicode files”.

Tabulator and blank line expansion

During transfers of text files, openFT carries out a tabulator and blank line expansion if necessary. This means that blank characters will be transferred instead of a tabulator, and a line with a blank character will be transferred instead of a blank line. During this, the following cases will be different for openFT partners:

Initiator	Direction	Responder	Expansion (yes/no)
Unix system, Windows system	Send	Unix system, Windows system	no, optional yes[1]
Unix system, Windows system	Fetch	Unix system, Windows system	no
Unix system, Windows system	Send	BS2000, z/OS	yes, optional no[1]
Unix system, Windows system	Fetch	BS2000, z/OS	no (not relevant)
BS2000, z/OS	Send	Unix system, Windows system	no (not relevant)
BS2000, z/OS	Fetch	Unix system, Windows system	yes (at the initiator)
BS2000, z/OS	Send and Fetch	BS2000, z/OS	no

[1]The expansion can be explicitly enabled or disabled in Unix systems and Windows system during the request.

During file transfer with FTAM partners, there is no blank line expansion. Tabulators are expanded during transfers using the character set *Graphic String,* but not in the *General String*. For more detailed information on FTAM character sets, see also section “FTAM files”.

Binary format

When “Binary format” is specified, it is assumed that the file to be transferred contains an unstructured sequence of binary data. In the receiving system, a file with an undefined record length (i.e. without end-of-line characters) is generated. The binary data remains the same.

User format

When sending a file, it is assumed that length fields divide up the file into records. The first two bytes of each record must indicate its length, including the length of the record length field. When the file is fetched, this length data is generated in accordance with the actual record lengths in the remote system. The contents of the records are treated like binary data, i.e. not converted.

Both the record structure and the binary data remain unchanged when a file is transferred. The record length fields are stored in all Unix and Windows systems starting with the most significant byte. The maximum permitted record length within a file in the user format depends on the operating system.

4.3.4 FTAM files

You can exchange the so-called “document types” FTAM-1 (for text files) and FTAM-3 (for binary files) with FTAM partners. This function is not available on z/OS.

The file structure and contents of these FTAM files are described in the Kernel group in “contents-type”:

- constraint set
  The constraint set describes the file structure. The subset of the FTAM standard selected by the functional standard ISO/EN 10607-3 permits only the value *unstructured*.The *constraint set* also specifies the actions which are permissible with the file on the basis of the structure of the file. For unstructured files, read, overwrite, extend and delete operations are permitted. Together with the *permitted actions*, the *constraint set* restricts the set of possible actions on a file.
- document type
  describes the actual contents of the file. ISO/EN 10607-3 requires support of FTAM-1 (unstructured text) and FTAM-3 (unstructured binary) for files with binary contents. The string format (*string significance*) can be variable (*variable*), fixed (*fix*) or not significant for storage (*not significant*). Furthermore, a maximum length of the string (*maximum string length*) can also be defined.

  In the case of text files (FTAM-1), the *universal class number* specifies the characters present in the text:

  - *GraphicString* can contain all graphical character sets (G sets) and escape sequences can be used to switch between character sets (see ISO 2022).

    openFT sets the character set to ISO 646 IRV (or ASCII IRV or ISO 8859-1 G0 set) plus ISO 8859-1 G1 set which broadly covers the characters used in the European languages. When two partners interconnect with openFT as of V10, the character set for file transfer is set to UTF-8.
  - *GeneralString* may contain not only graphical characters but also controlcharacter sets (C sets) which can also be switched.
  - *VisibleString* contains only graphical characters from ISO 646 IRV.
  - *IA5String* contains graphical characters from ISO 646 IRV and control characters from ISO 646 (C0 set).

4.3.5 Transfer of various file types

Besides complete transfer of the contents of a file, file transfer also aims at producing an authentic representation of the file structure. If identical structures are mapped to each other, as is the case with homogeneous links, authenticity is achieved without any problem, i.e. the binary code and the character representation are identical in the send and receive system. With heterogeneous links, however, it is usually not possible to obtain the binary code and the character representation in the receive system unchanged. For this reason, a distinction is made between text and binary transfer for file transfer with openFT. More details on file transfer with FTAM partners ca be found in the section “Special points for filetransfer with FTAM partners”.

Text transfer

Text transfer is character-oriented, i.e. the presentation of the characters is retained. This applies both to characters in single-byte code such as ISO 8859 and to Unicode characters which are represented by multiple bytes. The record structure of the text file is matched to the system conventions of the receive system when the file reaches the receive system.

The “useful data” of a file to be sent per text transfer must not contain any characters which the receive system could interpret as control characters, e.g. X‘15‘ (EBCDIC linefeed) and X‘0A‘ (ASCII linefeed).

> ¡ In the case of text transfers, openFT (z/OS) can use file-specific conversion tables that are selected via the file name; please consult your FT administrator.

The following table show the allowed settings for text transfer:

Record structure in receive system	**Local system**	**Remote system**	**Direction <-- / --> [1]**	**file type/ DATA TYPE**
system-conformant (in the usual manner in the receive system)	BS2000:DVS, PLAM	BS2000: DVS, PLAM	<-- / -->	Standard text binary user
	BS2000: DVS, PLAM, POSIX	BS2000: POSIX, Unix system, Windows, VMS, z/OS	<-- / -->	Standard text
	Unix system or Windows system	BS2000, z/OS	<-- / -->	Standard text
	Unix system or Windows system	Unix system or Windows system	<-- / -->	Standard text binary

[1] <-- = fetching, --> = sending

For details on z/OS, please refer to the manual "openFT (z/OS) - Command Interface".

Binary transfer

Binary transfer is carried out such that the coding (binary representation) of the characters is retained. The design of the record structure can be controlled. In this way, openFT matches the record structure with the record structure of the receive system (systemconformant record structure). With the original record structure, the structure of the send system is retained. Furthermore, it is possible to employ your own system-dependent record structures using the FT-specific user format.

> **i** It is not possible to fetch binary format files with fixed length or variable length records using the FTP protocol. In particular, this also applies to the output of file transfers with preprocessing on BS2000 or z/OS and the output from commands executed using *ftexec* on BS2000 or z/OS. In this case, you must either transfer files in text format or use a different transfer protocol (openFT).

The following table show the allowed settings for text transfer :

Record structure in receive system	**Local system**	**Remote system**	**Direction <-- / --> [1]**	**file type/ DATA TYPE**
system-conformant (in the usual manner in the receive system)	BS2000: DVS, PLAM	DVS, PLAM, z/OS	<-- / -->	Standard text binary user
	BS2000: DVS, PLAM	POSIX	<-- / -->	Standard text
	BS2000: POSIX	POSIX	<-- / -->	Standard text binary
	BS2000: POSIX	z/OS	<--	text binary
	Unix system or Windows system	Unix system or Windows system	<-- / -->	Standard text binary
original record structure (in the usual manner in the send system)	BS2000: POSIX	DVS, PLAM	-->	binary
	BS2000: POSIX	Unix system or Windows system	<-- / -->	binary
	BS2000: DVS, PLAM	Unix system or Windows system	<--	binary
	Unix system or Windows system	DVS, PLAM, z/OS	-->	binary
	Unix system or Windows system	POSIX, Unix system or Windows system, VMS	<-- / -->	binary

User format (system-independent)	BS2000: DVS, PLAM	POSIX	-->	user
	BS2000: DVS[2], PLAM, POSIX	Unix system or Windows system	-->	user
	BS2000: POSIX	Unix system or Windows system	<--	user
	Unix system or Windows system	DVS, PLAM, POSIX, z/OS	<--[3]	user
No record structure (i.e. the record structure is possibly lost)	BS2000: DVS, PLAM	POSIX, Unix system or Windows system, VMS	-->	binary
	Unix system or Windows system	DVS, PLAM, z/OS	<--	binary

[1] <-- = fetching, --> = sending

[2] applies only for SAM files with variable record length (RECFORM=V)

[3] A file can be sent in user format provided that the file has been fetched in user format

> Temporary BS2000 files cannot be transferred with openFT.
> ISAM and PAM files can be transferred between BS2000 systems and other systems as follows:
>
> - in transparent format, see "Transfer of various file types"
> - by specifying the target format, see the section “Heterogeneous transfer of PAM and ISAM files”

For details on z/OS, please refer to the manual "openFT (z/OS) - Command Interface".

Record by record transfer

Local Unix system or Windows system

When transferring files between Unix or Windows and BS2000 systems the structure of records can be important. If files are transferred from a Unix or Windows system to a DMS file, then you must increase the maximum record length if the block sizes generated by default for the DMS files are insufficient to accept the longest record. This is generally the case as of a net record length of 2024-2040 bytes.

Local BS2000 system

When DMS files are transferred between BS2000 systems, the file structure is not usually considered (the files are transferred block-by-block). The file’s record structure is significant in the following cases (the files are transferred record-by-record):

- transfer between BS2000 and Unix systems, Windows or z/OS
- extension of a file with a record structure
- transfer of POSIX files
- transfer of library members

In these cases, the maximum length of the records that are to be transferred should not exceed the following values:

- Partner systems with openFT V10 and higher:
  - 32768 bytes in files with fixed-length records
  - 32760 bytes in files with variable-length records
- Partner systems with openFT < V10:
  - 32760 bytes in files with fixed-length records
  - 32758 bytes in files with variable-length records
  - 32248 bytes for compressed transfer (COMPRESS = *BYTE)

If files are transferred from Unix systems, Windows or POSIX to a DMS file then the maximum record length without the specification of the RECORD-SIZE parameter in the command may not exceed (2048*b-n) bytes, where b is the blocking factor of the BS2000 receive file. Default on NK-PVS disks is b=2, otherwise the default is b=1.
On K-PVS (K for Key) n=8, and on NK-PVS (NK for Non Key) n=20.

Local z/OS system

openFT (z/OS) usually takes into account the record structure during file transfer (exception: transfer of an unstructured sequence of binary data during file transfer with Windows and Unix systems). With record-by-record transfer the maximum length of the records to be transferred may not exceed the following values:

- 32760 byte in files with fixed-length records
- 32752 byte for files with variable record lengths (record length without record length field)
- 32248 byte for compressed transfer (COMPRESS = *BYTE)

Transfer with transparent file format

A special case is the transparent file format. This file format provides you with the option of passing through any BS2000 files over a variety of FT platforms to a BS2000 system, while retaining their original file attributes. This procedure is useful for distributing BS2000 files from a Unix based server or Windows server to BS2000 systems, for example. From the point of view of the intermediate processor, the files received, which cannot be used by this processor, are binary files. These files are then set up on the receive processor with their original attributes by openFT (BS2000).

Specific features in z/OS

openFT (z/OS) can act as a buffer for BS2000 files in transparent file format. However the transfer of these files must be initiated in the BS2000.

openFT does not provide any direct means of transferring z/OS files true-to-format (“transparent”) over FT platforms other than z/OS. However, you can use the TSO command XMIT to pack files in a neutral format and transfer them in this format as binary files with a fixed record length of 80 bytes. You do this, for example, by specifying ***-r=f80*** in the openFT *ft* command in Windows or on a Unix system. The file can then be unpacked at the target system using the TSO command RECEIVE.

Heterogeneous transfer of PAM and ISAM files

You can transfer BS2000 PAM files onto a foreign system such as a Unix or Windows system or to z/OS and then retrieve them to BS2000 and store them there as PAM files. The foreign system can also have the initiative for this request. You can also transfer ISAM files from a BS2000 systems onto a foreign system. In all cases, the prerequisite for this is that openFT as of V11 is running on the foreign system.

To do this, proceed as follows:

- Transferring a PAM file from BS2000 to a foreign system
  Specify "sequential" as the target format in the transfer request. The file content is transferred without the end-of-file marker of the C runtime system and empty blocks are initialized.
  The content of the PAMKEY and the architecture of the pubset are not transferred to the target system.
- Storing a binary file from a foreign system as a PAM file in BS2000
  Specify "binär" as the file format and "block-structured" as the target format in the transfer request. All blocks apart from the final one are completely filled with data and the end-of-file marker of the C runtime system is appended at the end.

  i
  - If you transfer PAM files out of BS2000 and fetch them back again, the original pubset and the target pubset should, wherever possible, have the same block properties. If this is not the case, or if the content of the PAM keys is important for the file, you have to assume that the target file will become unusable. In the case of PLAM libraries, it is generally possible to retain the format during transfer. If the target pubset is NK4, the PLAM library must be converted with PAMCONV.
  - If a PAM file that has been swapped out is to be transferred back to a BS2000 system with openFT V10, it is stored as a sequential file with an undefined record format. The request is rejected under openFT with a Version < V10.
  - It is not possible to extend a PAM file by fetching a file from the foreign system.

- Transferring an ISAM file to the foreign system
  Specify "sequential" as the target format in the transfer request. The ISAM keys are integral parts of the records that are read and are therefore transferred with the file. However, they no longer have any function as index keys. The record format of the target file is to be the same as that of the ISAM file. The format used is compatible with FTP-BS2000.

4.3.6 Transferring directories (Unix and Windows systems)

On Unix and Windows systems, you can transfer a complete directory between partner systems using a single request. You have the following options:

- Synchronous and asynchronous sending
- Synchronous and asynchronous fetching
- „Create new“ and „Overwrite“ mode

Restrictions: Directory transfer is supported only for openFT partners. It is not possible to extend existing files or to perform preprocessing and postprocessing.

4.3.7 Migrated files

openFT can access migrated files in BS2000 and z/OS systems. This means that you can view the properties of such files, and transfer, delete or overwrite them. To do this, openFT as of V10 must be used in the system involved. The following applies to the mainframe systems used:

- In BS2000 systems, the file must be a DMS file. It is not possible to directly transfer individual elements of a migrated library. To do this, the migrated library must first be read in. This can, for instance, be done during preprocessing and postprocessing or using /EXEC-REM-CMD or *ftexec*.
- In z/OS systems, z/OS as of V1.7 must be used, because the necessary values are only returned at the system interface as of this version.

4.4 Transferring 7-bit, 8-bit and Unicode files

In computers with different operating systems, the individual characters, letters and digits are represented internally ("coded") in different ways. In addition, it is possible to use different character sets in these various systems. The content of a text file is interpreted differently depending on the character set used and is output accordingly on the screen or at the printer.

Different character sets are of significance when transferring text files, but not when transferring other file formats (binary, transparent, etc.), as openFT does not convert the contents of the file in this event.

Code conversion of text files is performed by openFT.

openFT makes it possible to assign various single-byte character sets (7-bit and 8-bit) as well as multi-byte character sets (Unicode) to text files.

> **Note for z/OS:** A complete PO/PDSE data set is not converted in the sending system with openFT because otherwise it is possible that the control information in the destination file may no longer be correct.

4.4.1 Code tables and coded character sets (CCS)

The concept of so-called “Coded Character Sets” (CCS) is supported for openFT partners as of V10. A CCS defines a character set and the coding of these characters in the file. A CCS is assigned a name of up to 8 characters in length via which the CCS can be addressed.

When transferring text files, users can specify a separate CCS for file encoding in the local and remote systems.

Frequently used Coded Character Sets are:

ISO88591	Character set in accordance with the definition contained in ISO standard 8859-1, ASCII-oriented coding in accordance with ISO standard 8859-1.
EDF041	Character set in accordance with the definition contained in ISO standard 8859-1, EBCDIC-oriented coding in accordance with Fujitsu definition DF04-1.
IBM1047	Character set as defined in ISO 8859-1. IBM1047 is an EBCDIC-based encoding compliant with the IBM definition IBM1047 and used as default in z/OS systems.
UTF8	The character set is Unicode, the UTF-8 multi-byte coding defined in the Unicode standard is used.
UTF16	The character set is Unicode, the UTF16 16-bit coding defined in the Unicode standard is used.
CP1252	The character set is a superset, defined by Microsoft, of the character set defined in ISO standard 8859-1. The ASCII-oriented coding is identical to the ISO8859-1 for the characters which are shared with ISO8859-1. The other characters defined by Microsoft (including the Euro symbol) are present in the code range 0x80-0x9F which is not used by ISO8859-1.

Making a CCS available

- In BS2000 systems, the CCSs are defined and made available via XHCS. The default CCS for the system (HOSTCODE) is defined by the BS2000 system administrator. The administrator can also assign a default user character set different to HOSTCODE to a user ID.
- On the other openFT platforms as of V10, the commonly used CCSs are supplied with openFT. The FT administrator defines the default character set via the operating parameters.

4.4.2 Specifying the CCS on a transfer request

When transferring text files, you can specify a request-specific CCS for both the local system and the remote system which is used for reading or writing the local and/or remote file.

If the local or the remote file is a BS2000 file to which a CCS name has already been assigned via the catalog entry then you may not specify a CCS name that is different from this.

The remote CCS name is only supported for the openFT protocol and for partners as of V10.

If the local or remote CCS name is omitted then the default settings for the relevant system apply:

- openFT operating parameters in a Unix system, Windows system or z/OS system,
- in a BS2000 system, the CCS corresponding to the file's catalog entry (if present), otherwise the HOSTCODE system parameter.

In z/OS, a particular CCS can be assigned to files on the basis of a setting in the FT parameter library. For details, please refer to manual "openFT (z/OS) - Installation and Operation"

> **Caution!**
>
> If you save the file in a character set which is not a superset of the character set originally used for the file then information is lost! All characters that cannot be mapped to the newly assigned character set are represented by a replacement character. This type of conversion cannot be undone without data loss!

4.4.3 Data conversion

The type of data conversion depends on the openFT version that is used on the partner system.

Data conversion in the case of partners as of V10

Depending on the code class (ISO 8859 or DF04) and code variant n (n=1...10, 13, 15) of the local CCS, openFT as of V10 sends the data encoded in ISO 8859-n, DF04-n or UTF-8.

This has the following effect depending on the partner system and the encoding:

- Files in Unix and Windows systems to which an ISO8859n CCS is assigned are no longer recoded in the event of send requests to Unix or Windows systems. In the case of transfers between Unix or Windows systems no recoding is now performed for the transfer itself if the same ISO8859n CCS has also been assigned for the target file.
- In the case of transferring files belonging to the code classes ISO 8859 or DF04 between Unix and Windows systems and BS2000 or z/OS, recoding is performed at the receiving system (if necessary).
- UTF-8 files are recoded at the receiving system (if necessary). Files to which a CCS is assigned that belongs neither to the ISO 8859 code class not to DF04 are recoded into UTF-8 at the sending system and into the CCS of the target file at the receiving system (if necessary).
- UTF-16 files are recoded into UTF-8 at the sending system and into UTF-16 at the receiving system (if this is requested).
- UTF-16 files generated by openFT possess the endian model and line break convention (LF or CRLF) appropriate to the platform in question.
- UTF-8 files generated by openFT possess the line break convention (e.g. LF or CRLF) appropriate to the platform in question.

Data conversion in the case of partners < V10

The transferred data is coded in DF04-n. I.e. when file transfer is performed with openFT partners, the data is transferred in EBCDIC format (corresponds to CCS DF04-n). EBCDIC is used, for example, in BS2000 systems. In the case of transfers with openFT partners on Unix or Windows systems, text files are recoded in the partner system.

Notes for Unix and Windows systems

openFT converts text files when transferring to and from openFT partners as follows:

- when retrieving a file into a Unix or Windows system from EBCDIC to ISO 8859,
- when sending a file from a Unix or Windows system from ISO 8859 to EBCDIC.

Special characters or alternate representations not defined in ISO 8859 are not converted during code conversion. Files containing such characters should be transferred as binary files, and converted using a user-defined code conversion routine.

In the case of data transfer handled using the FTAM functionality, it is assumed that ISO 8859 is used for the transfer and for the local file with connections between third-party products and openFT partners < V10. No local recoding is therefore performed.

Text format

When sending, openFT assumes that the file to be sent is a pure ISO 8859 text file, which is structured as records separated by carriage returns/line feeds.

In certain situations, a conversion takes place, i.e. tab characters are expanded into blanks and end-of-line characters are eliminated. Depending on the situation (inbound, outbound) and the participating partners, the following applies:

- Inbound requests:

  Conversion to Unix or Windows is not available for send or receive operations on the inbound side.
- Outbound requests issued by a Unix or Windows system:

  Conversion never occurs when receiving requests.

  Request-specific conversion *(ft -tb=* and *ncopy -tb=*, TabExpansion) is possible on send operations. By default, send operations to BS2000, OS/390 or z/OS partners are converted. In all other cases conversion does not take place.
- Outbound requests which are issued in a BS2000, OS/390 or z/OS system:

  Conversion never occurs when sending requests.

  Conversion occurs when receiving requests, depending on the partner, i.e. conversion occurs for a Unix or Windows partner but not for BS2000, OS/390 or z/OS partners.

Binary format

openFT assumes that the file to be transferred contains an unstructured sequence of binary data. In the receiving system, a file is created with an undefined record length. The binary data is retained.

User format

When sending, openFT assumes that the file to be sent is structured by length fields in records. The first two bytes of each record must contain the length of that record, including the length of the record length field. When retrieving, openFT generates these length specifications in accordance with the record lengths in the remote system. The record contents are handled as binary data, i.e. not subjected to code conversion.

The record structure and the binary data are retained during transfer. The highest-order byte of the record length field is stored first in a Windows system.

> ¡ There is no point using user format for FTP partners since the record structure is lost. A different mechanism is used between FTAM partners (see section “Virtual filestore”).

4.4.4 XHCS support by openFT on BS2000 systems

With XHCS, various codes can be used in a BS2000 system at the same time. openFT can utilize XHCS information to recognize the code being used. Depending on the type and scope of the information, openFT (BS2000) employs the XHCS conversion tables (either before or after a file is transferred as a text file) to convert a file to a code that can be processed in the target system.

For details, please refer to manual "openFT (BS2000) - Installation and Operation".

4.5 Character code support for file names and commands on Unix and Windows systems

On Unix and Windows systems, openFT supports different character coding for the specification of

- remote file names and directory names,
- pre, post and follow-up processing,
- remote commands

A distinction is made between the **transparent mode** and the **character mode** for these parameters

Transparent mode

file names, etc. are seen in a fixed binary code, independent of local character code settings. Code transformation merely takes place between EBCDIC DF.04-1 (BS2000), IBM1047 (z/OS) and ISO8859-1 (Unix, Windows). If the systems involved use different code variants to present their file names, the characters may then change. Transparent mode is compatible with the previous versions of openFT.

Note on Windows systems

In Windows there is a certain dependence on the locally set ANSI code for remote file names and command strings in remote follow-up processing in transparent mode, because file names and commands were seen in local ANSI code in earlier openFT versions at the Windows system interfaces. Regardless of the selected mode this dependence also exists for most other parameters, such as remote access authorization.

Character mode

File names, etc. are seen in their character presentation. An Ä in a remote file name is for example understood in character mode in the partner system also as an Ä, even if different system coding is set there and different local coding is set. If the Ä is not included in the character set of the partner system, the command is aborted (e.g. "Syntax error in the resulting file name"). The character mode assumes that the partner is openFT V12.1 or higher.

The FTP protocol does not support the character mode.

Note on Windows systems

In Windows systems it is possible in character mode to specify remote file names and command strings in remote follow-up processing regardless of the locally set ANSI code; the character set - compared with that of Windows file names (UCS-2) - is not restricted. This even applies for local file names regardless of the selected mode.

4.5.1 Properties of the character mode

The transfer of path names between the partner systems takes place in UTF-8 in character mode so that characters outside the scope previously supported by openFT can also be transferred and thus exotic path names can also be addressed.

Displaying remote file names

If a command for the reading of file information is executed in character mode, the file names obtained from the partner system are then also converted from the character code of the partner to the local character code. Files, whose names do not correspond to a valid coding according to the setting in the partner system, are skipped when listing directories. For example, file names from a Unix system are not shown with a set UTF-8 (*ftmodo -fnccs=utf8*) when they do not correspond to a valid UTF-8 coding. In transparent mode on the other hand file names are transformed according to the previous regulations of openFT and according to the operating systems involved and are retained as byte strings. File names on Windows, which do not correspond to the character set of the set ANSI code, are skipped.

Note on Unix systems

The character code of a Unix partner system is determined on the inbound side from the new openFT option FNCCS, which the openFT administrator of this system sets via operating parameter (e.g. using the command *ftmodo -fnccs*). For this purpose he should specify the predominant local coding, e.g. UTF8 or ISO8859-1. On the outbound side the character presentation for the specification of remote file names, pre-processing commands, etc. is derived from "LOCALE" and the environment variable LANG (e.g. ISO8859F for de_DE@euro, ISO88597 for el_GR, or UTF8 for de_DE.utf8). If commands are entered via the terminal or console, the terminal display should also correspond to this setting. The code table derived from the "LOCALE" and LANG variables must be integrated in openFT. Character codes, in which the coding of the ISO646 character set differs from the ISO8859-1 coding (e. g. ISO646de, EDF041, UTF16), should be avoided both for the definition of FNCCS and the LANG setting. Otherwise, pre-processing, ftexec and ftadm, or even normal file accesses would not work correctly in character mode.

Note on Windows systems

In the current Windows systems the path names are always stored in the file system in UCS-2 (a subset of UTF-16) so that no further openFT option is necessary here. Path names are coded in UTF-8 in the request description, and in character mode also for transfer.

Note on BS2000 and z/OS systems

Fixed character codes are also assumed (EBCDIC DF.04-1 and IBM1047) in character mode in BS2000 and z/OS.

4.5.2 Recommendations for the use of transparent and character mode

If ISO646 characters are always used for file names and follow-up processing, it is then possible to work both with transparent and with character mode. Otherwise please read the following sections.

4.5.2.1 Using the transparent mode

- If the character set of file names is restricted to ISO8859-1, and if appropriate code settings are also used on the computers (e.g. LANG=de_DE on Unix and CP1252 on Windows), transparent mode is recommended. If the character set is restricted to a code variant of ISO8859 (e.g. ISO8859-7 with Greek characters) for an application of the openFT product, it is then possible to work with transparent mode.
- As long as there are no problems with the character presentation of file names (e.g. due to the expansion to additional language areas), it is recommended to retain transparent mode in existing applications of openFT.

> ℹ If instances from several language areas are coupled with each other in an application of openFT, which in each case use national code settings (e.g. ISO8859-1 and ISO8859-7), the character presentation of file names may vary if transparent mode is used.

4.5.2.2 Using the character mode

- If an extension to the file name character set of ISO646 to an 8-bit character code is imminent, a switch to character mode should be considered as a matter of foresight, as this can be done more easily from ISO646 than if a switch has been made in the meantime to an 8-bit character code.
- If UTF-8 is used as character code for file names in the instances of an openFT application, it is recommended to select character mode. Character mode can be most effectively used if all instances code their file names in UTF-8 (this already applies in principle on Windows instances thanks to the use of character mode).

> **Caution!**
>
> If instances from several language areas are coupled with each other in an application of openFT, and if "national" or "regional" code settings, such as ISO8859-1, are used for file names in specific instances instead of UTF-8, file transfer requests in character mode, whose specification cannot be mapped for the remote file name in the code set there, are rejected.

Notes on Windows systems

- The change to character mode is not a problem for file systems on Windows.
- File names in the output of commands, such as *ftshwl*, *ftshwr*, *ftshwp* etc. are represented in Unicode on Windows systems. However, *ftshwl | more* replaces characters outside the locally set code table with ? or other substitute characters.
  It is advisable for larger output to set OPENFTOUT=UTF8 as the environment variable, to reroute the output to a file and then to look at it with *ftedit -ro -ccs=utf8*.
- If openFT on Windows creates files on a directory, which is not on a local drive (e.g. a directory connected via SAMBA), this can result in character replacements in the file name, which would occur during the local copying of a Windows file to this directory.

Notes on Unix systems

- Care is required in the case of a change in character coding for file names on a Unix system, for example when changing from transparent mode to character mode. This can result in inconsistently coded file trees (such as UTF-8 coded file names mixed with ANSI file names in an old directory, which also contains an ANSI umlaut in the name), with which not only openFT is likely to have difficulties.
  All file and directory names, references to file names, etc., which are still to be accessed after the change, would have to be renamed to suit the new coding (e.g. ISO8859-1 -> UTF-8). Only if ISO646 characters have until now been exclusively used in file names, is a switch possible without this additional effort.
- If openFT on Unix only accesses inbound files for an application, which are in a defined directory, this can be specified as a file name prefix in an admission profile. If character mode openFT requests use the access authorization of this admission profile, it is sufficient that the file names relative to this prefix correspond to the code setting; there are no limitations in this case for the directory name. For example, coded umlauts would also be allowed in ISO8859-1, even if *ftmodo -fnccs=utf8* is set.

Transfer admissions

Care should also be taken when defining transfer admissions with characters outside the ISO646 character set. In the case of file transfer requests and file management requests the specified transfer admissions is always processed according to transparent mode - even if character mode was selected (for file names). However, when specifying transfer admissions on a local Unix, as well as via remote administration in character mode, this is created in the local character code set there (in the case of remote administration in character code, which is defined with *ftmodo -fnccs=...*).

Trace evaluations

Trace evaluations can also contain UTF-8 strings in file name specifications. It may thus be expedient to look at an evaluated trace with *ftedit -ro -ccs=utf8* in order to see these file names in their correct character presentation. Byte sequences in the trace evaluation, which do not correspond to the correct UTF-8, are nevertheless not suppressed or replaced with ?, _ or the like, but are evaluated via heuristics as a ISO8859-1 character. As a result, the information as to whether an umlaut is coded in ISO or in UTF-8 is lost. In order to establish this it makes sense to look at the trace without the switch *-ccs=utf8*, too.

4.6 Entries for the remote system

With the entries for the remote system, you define the partner system and inform it of your transfer admission for a login name in the partner system.

For details on the partner concept and on partner systems, please refer to chapter “Partner concept”.

4.6.1 Defining the partner computer

The partner system is the remote system with which files are to be exchanged. You address the partner system via a partner name or its partner address ("**dynamic partners**").

The FT administrator may deactivate the use of dynamic partners for security reasons. In this case, you may only use partner names from the partner list.

Partner name

A partner name is a name of 8 characters or less which is assigned by the FT administrator when including a partner system in the partner list. This approach should primarily be used for partner systems which are frequently communicated with.

Partner address

If the FT administrator has not assigned a partner name or if you do not know the name, you can address a partner host using the partner address. A partner address has the following structure:

[protocol://]host[:[port].[tsel].[ssel].[psel]]

host (= computer name of the partner system) is mandatory; all other specifications are optional, see also section “Partner addresses”. In many cases, the other specifications are covered by the default values, so that the host name suffices as the partner address.

The individual components of the address are explained in the corresponding openFT manuals „Command Interface“

4.6.2 Transfer admission

The transfer admission can be specified as:

- login/LOGON admission containing the login name, the account number and the password. These values are system-dependent.
- FTAC transfer admission with an operating system-independent definition which provides a higher degree of access protection. The FTAC transfer admission has to be defined in the partner system.

The Inbound access via the default FTP client is also possible.

Further details are explained in the corresponding openFT manuals „Command Interface“.

4.7 Options for file transfer

openFT offers the possibility to make additional optional setting for file transfer. You can define individual record lengths, agree write mode and file compression, and specify conditions for result messages and access modalities for FTAM partners.

4.7.1 Maximum record lengths

The maximum record length is understood to be the length of the longest record (net record length in bytes) not including the record length fields.

Because openFT can also transfer files using unicode character sets one must distinguish between length in bytes and length in characters.

In BS2000 and z/OS files, the maximum record length is stored as a file attribute in the catalog (with variable-length records and an additional allowance of 4 for the record length field).

In Unix systems, Windows systems and POSIX systems, you can set the maximum length of your file which you wish to transfer as text or record-structured binary file (user format) individually. The prescribed maximum record length must be at least as large as the largest one actually available, otherwise the FT request cannot be executed.

Please note that the maximum record length must be observed not only on the sending and receiving system but also during transfer. In the case of a transfer in the UTF-8 character set (see section “Data conversion”), the record length needed during transfer may be larger than the record length on the sending or receiving system.

4.7.2 Write mode

With the option “Write mode”, you can define the procedure to be adopted for the destination file during file transfer. This option can also be defined via FTAC. There are the following options:

- to overwrite files, i.e. files are overwritten, provided that the file attribute permit this action, or files that do not exist are created,
- to extend files, i.e. existing files are extended at the end of the file, provided that the file attribute permit this action, or files that do not exist are created,
- to not overwrite files; in this case, existing files are under no circumstances overwritten; rather, the FT request is aborted with an appropriate message. If the specified destination file does not exist, a new file is created.

In z/OS, the precise effect of the "Write mode" option (WRITE-Mode operand) also depends on the type of receive file (PS data set, member of a PO data set, etc.). This is described in detail in the manual "openFT (z/OS) - Command Interface".

4.7.3 Compressed file transfer

Files can be sent using data compression. This shortens transmission times and saves costs. However, do note that compression and decompression produce extra CPU load in the receive processor.

openFT is able to use two compression methods - zip compression (with openFT partners as of V10) and byte compression. Both of these can be used to reduce the volume of data for transfer. However, compressing and decompressing the data increases CPU demand and consequently also the time required for a request before and after data transfer itself.

On "fast" lines (as of approximately 10 Mbit), the overall execution time of a request normally is not significantly improved by compression. On "slow" lines (less than1Mbit), zip compression may help enhance performance. Byte compression is worthwhile when transferring files which contain a large number of byte repetitions (e.g. lists with blanks for column alignment, dumps with numerous zeros). If the partner does not support compression, openFT transfers the file uncompressed. openFT-FTP supports byte compression as described in RFC959.

Data compression is not supported on transfers to FTAM partners.

4.7.4 Encrypted file transfer and encrypted file management

openFT can send data with encryption if requested by the user (see also the section “Encryption of the file contents”)

For file management requests, the user can require the encryption of file and directory list attributes, see section “Encryption for file management requests”.

For legal reasons, the encryption option is not available in all countries, i.e. the encrypted file transfer and encrypted file management with foreign partners is not guaranteed in all cases.

Data encrypted by openFT can only be exchanged via the FTP protocol in an outbound direction and only with secure FTP partners. No data encrypted by openFT can be exchanged with FTAM partners.

Encrypted file transfer and encrypted file management always require openFT-CR to be installed on the openFT side, i.e. also on the partner system if openFT is running there.

4.7.5 Transfer of protection attributes between BS2000 systems

If the partner is a BS2000 system running openFT as of V11.0, DMS files can be transferred in such a way that the protection attributes USER-ACCESS, ACCESS, BASIC-ACL, EXPIRATION-DATE, FREE-FOR-DELETION and DESTROY are also transferred in addition to the default file attributes.

The protection attributes can only be transferred via the openFT protocol and they cannot be transferred in transparent mode. The target file must be created as new or overwritten and must not be a file generation.

4.7.6 Notifying results

The initiator of a file transfer request can arrange to be notified of the result. The logging function, which is available in a standard form on all platforms, is particularly suitable for this.

Other ways of notifying results are platform-dependent:

- On BS2000 systems, the FT system sends asynchronous messages to the user which provide information on the result of the file transfer. Prerequisite is that the user task submitting the file transfer request is still active and that asynchronous messages are allowed.
- On z/OS and BS2000 systems, a file is created on request by the initiator and can be printed out automatically on success or failure of the file transfer.
- On Unix systems, the result message can be stored in the mailbox of the initiator depending on the result.

For details on z/OS, please refer to the manual "openFT (z/OS) - Command Interface".

4.7.7 Access mode for FTAM partners

It is possible to define FTAM-specific file attributes for file transfer with FTAM. The FTAM file attributes that describe the file type must be identical to those specified in the file transfer request. The corresponding attributes are presented in the section “Special Points for file transfer with FTAM partners”.

4.7.8 Preprocessing and postprocessing

The “preprocessing” and “postprocessing” functions make it possible to execute any commands (operating system commands, procedures, etc.) with the aid of a file transfer request in the local and remote systems. The commands are passed to the corresponding system instead of the file name. To do this, the following applies:

- the file name must be enclosed in double quotes (Unix systems and Windows systems) or must be specified as a C string (BS2000, z/OS).
- The first character is a pipe symbol ’|‘. Then follow the commands, separated by ';' (or '&' or '&&' in Windows systems, in which case the command string must start with *cmd /c*). The maximum length of the pre- and postprocessing command is limited by the maximum length of the file name.

  If the characters ’|&’ are specified instead of the pipe symbol, the transfer request is restartable, see "Preprocessing and postprocessing".

Preprocessing passes the result to the system’s standard output (SYSLST on BS2000, SYSPRINT on z/OS, stdout on Unix systems and Windows systems). Postprocessing reads the data from the relevant system’s standard input (SYSIN on BS2000, SYSTSIN on z/OS, stdin on Unix systems and Windows systems). However, the standard output/input does not usually support all the file formats possible at the system in question. You can avoid this restriction by using the %TEMPFILE variable instead of the standard output/input. This has the advantage of permitting the use of any required file format. Even if a preprocessing command cannot be output to the standard output or if a postprocessing command cannot read from standard input, normally it may be helpful to specify % TEMPFILE in the request parameters.

Pre- and postprocessing are part of the request brackets. The issuer of the request always receives a feedback report on the successful or unsuccessful completion of the pre/postprocessing.

You should note the following when using the pre/postprocessing function:

- Preprocessing/postprocessing runs as part of the file transfer operation and under the same transfer admission. These specifications are either explicitly stated in the file transfer request or in a transmission profile’s USER-ADMISSION.
- If the request is handled via an admission profile, the FILE-PROCESSING function must be permitted in the profile or, alternatively, a file name prefix starting with the pipe symbol '|' must be defined.
- When non-restartable pre/postprocessing is involved, the connection to the partner must remain intact until the entire processing session is completed.

Restart capability during preprocessing and postprocessing

During restartable pre- and postprocessing, the data to be transferred between openFT and the processing command is always saved to a temporary file. By this means, the request is divided into 3 phases: preprocessing, transfer and postprocessing.

The restart capability of a pre- and postprocessing session is brought about when you specify an additional “&” before pre- and postprocessing in the transfer command. During this, requests made with openFT partners behave as follows:

- Loss of connection during preprocessing:
  If the connection is lost during the execution of the preprocessing command, the command is still executed until completion after the connection is lost. If the system is restarted after the command has completed execution, then the temporary file is transferred.

- Loss of connection during transmission:
  In this case openFT performs a restart for the temporary file as is usually the case.
- Loss of connection during postprocessing:
  If the connection is lost during the execution of the postprocessing command, the command is still executed until completion after the connection is lost. If the system is restarted, then all other actions left over that belong to the openFT request are performed (e.g. any follow-up processing or the status report to the partner).

Notes on BS2000 systems

The openFT subsystem cannot be stopped, as long as there are still restartable requests, whose pre- and postprocessing sessions have not been completed. If this is taking too long, the system administrator can cancel the batch jobs belonging to the requests (using the CANCEL command). In this case, the message FTR2083 is displayed during the next restart of openFT.

If restartable requests are still active in the command execution phase when openFT is stopped using STOP-FT, then shut-down is delayed by up to 2 minutes. If command execution still has not been completed after 2 minutes, openFT will be stopped and the request will be cancelled during the next START-FT.

Notes on Unix systems and Windows systems

The temporary file is stored in the directory *.openFTTmp* and is deleted only after the command has finished execution (regardless of whether or not the command was successful or unsuccessful).

.openFTTmp is created by openFT if it does not yet exist. It is located in the home directory of the corresponding user. On the local host this user is the user under whose user ID the request was started. On the remote host this user is the user whose user ID was specified or who is the owner of the specified transfer admission.

> **i** If there are still restartable requests active when an openFT shutdown is initiated and they are still in the command execution phase, then the shutdown is delayed for up to 10 minutes so that the commands have enough time to execute to completion. During this period, a command to shut down the openFT server remains "pending" and the prompt is not displayed until the server process has terminated.

Server function for remote command execution (ftexec)

One special form of preprocessing is the server function for the remote command execution (*ftexec* command for Unix and Windows systems, EXECUTE-REMOTE-CMD command for BS2000 and FTEXEC command for z/OS). This command makes it possible to execute commands on a remote system. The exit code and/or the output from *stdout* and *stderr* (Unix or Windows systems), SYSLST and SYSOUT (BS2000) or STDOUT=SYSPRINT und STDERR=SYSTSPR (z/OS) are output at the local computer. This command thus acts as the execution of the command on the local computer.

4.7.9 Follow-up processing

openFT offers four types of follow-up processing requests:

- Follow-up processing in the local system after successful file transfer
- Follow-up processing in the remote system after successful file transfer
- Follow-up processing in the local system after unsuccessful file transfer
- Follow-up processing in the remote system after unsuccessful file transfer

The conventions of the system on which the follow-up processing is to be performed are decisive for the syntax and processing of the statements and commands. A command sequence can only be processed in the remote system if an FT that supports this function is used in the remote system.

Variables for follow-up processing

You may specify variables within the command or command sequence for follow-up processing, e.g. for the partner name oder the file name. These are substituted at the start of follow-up processing in the particular system using the values obtained from the file transfer requests.

Details on the allowed variables can be found in the corresponding openFT manual "Command Interface".

Maximum length for follow-up processing

You may specify data for follow-up processing both for the local and for the remote system, depending on the of openFT version used. In each case, no more than 1000 characters may be used. The number of characters evaluated depends on the operating system and is stated in the relevant FT description. Please observe that

- the limit length applies after any necessary translation of variables.
- as of openFT V12, follow-up processing commands in Windows systems are converted into the UTF-8 character code and that therefore characters that are not present in the ISO646 character set occupy more than one byte in memory.

The limit of up to 1000 characters can be bypassed by calling a procedure, a shell script or a program from within the follow-up processing. A procedure may contain the command sequence which is to be executed on success or failure of file transfer.

Follow-up processing with FTP or FTAM partners

Restrictions apply to transfers with FTP or FTAM partners, since the FTP or FTAM protocol does not permit transfer of follow-up processing data. Follow-up processing in the FTP or FTAM partner system is possible only if it is stipulated there in an FTAC admission profile. It is always possible to initiate follow-up processing in the local system.

Deleting the send or receive file following the file transfer

The special form of follow-up processing, *DELETE, is available for requests on which the send file is to be deleted following successful transmission. On the local system, this function is available for all partners. On the remote system, this function is only available for openFT and FTAM partners.

To avoid undefined file fragments in the event of unsuccessful file transfer, it is useful to delete the receive file via follow-up processing in such cases.

4.8 Access protection for send and receive files

openFT without FTAC functionality offers the same transfer and access protection as the operating system. The FT user must identif himself as authorized for access to a file via the FT system in the same manner as for the file management system of the operating system. This means that a complete LOGON/login authorization as well as any file password required, must be given.

The use of openFT with FTAC functionality is an extension of the transfer and access protection features of the operating system to include the security mechanisms contained in the FTAC functionality.

4.8.1 Access protection during transfer

Please note that the destination file is generally not protected from being overwritten by other users while the time the request is being processed. If the transfer is interrupted, for example, then other users may be able to write to the destination file. Access protection differs in the individual systems:

BS2000 systems

openFT (BS2000) uses a file lock which protects the files if the transmission is interrupted and between the time of accepting and processing the FT request. This protection does not apply to library members and POSIX files.

When a file transfer request is accepted, a lock is set on each file to be transferred. Only read access is granted to other users for the send files; no access is permitted for the receive files. The BS2000 command SHOW-FILE-LOCK indicates whether a file has been locked by openFT and lists the transfer ID's involved. File locks are automatically removed on unloading the subsystem.

The mode of operation of openFT dictates that a receive file which already exists can only be overwritten if both read and write access are available for this file. For file access, the specifications of the ACL (Access Control List) and BASIC-ACL must be adhered to.

The following table shows the conditions under which the FT user can access a BS2000 file:

Access mode	Conditions for file access
Read of a sending file	• File cataloged under specified login name or • File defined as multiuser or • User is working under the login name TSOS and • Write access is permitted and • Valid password was given, when the file to be transferred is read and execute protected by a password
Overwrite an existing receive file	• File cataloged under specified login name or • File defined as multiuser or • User is working under the login name TSOS and • Read and write access is permitted and • Valid password was given, when the file to be transferred is readprotected by a password

z/OS systems

openFT (z/OS) protects send and receive files against simultaneous (write) accesses only if data is in fact being transferred, i.e. if the request is in the ACTIVE state. It follows, that the send and receive files are not protected, if the file transfer has not yet begun or has just been interrupted.

If openFT attempts to access a send or receive file which is locked (for example, because another FT job is already accessing it), the FT job is rejected or terminated.

For a member of a PO or PDSE data set, this means:

- When a member of a PO or PDSE data set is to be read (send file), no other member of the same data set may be open for writing or only for reading when the request is issued, nor be opened before the end of the file transfer.
- When a member of a PO or PDSE data set is to be written (receive file), neither another member of the same data set, nor the data set itself may be open when the request is issued or opened before the end of the file transfer. In this case (receive file), even the display of the member list can cause the FT job to be aborted (for example when a send request is started from the menu interface, see manual "openFT (z/OS) - Command Interface", or the use of the PDF function "member list" in general).

  When an FT request is aborted because of an attempt to access a locked file, an error message is output.

Other systems

In other systems, for example Unix and Windows systems, or even in BS2000 systems in the case of POSIX file or library elements, the user is solely responsible for guaranteeing exclusive access to the files to be transferred . In these systems, the file cannot be occupied exclusively by openFT, not even during file transfer.

The user him/herself must therefore ensure that (the data and file attributes) in the file to be transferred are consistent throughout the entire duration of the FT request. This applies to both the send and receive files. The danger of eventual inconsistencies resulting from multiple accesses can be reduced, for example, by means of access restrictions (Unix system: *chmod* command). It is also possible to transfer the file to a different name or to a temporary directory and to rename it or move it to a different directory only after file transfer has been completed successfully using follow-up processing.

4.8.2 Specific features in z/OS

The software products SYS1.UADS and RACF (or compatible products like TOP-SECRET and ACF-2) installed in the z/OS system are used to check the transfer and access admissions of the FT user. Therefore, the same conditions to read and write file access apply to openFT and TSO or JES2-/JES3 users.

For send files and existing receive files, openFT uses the products named above to check the FT user's access rights (read/write) against the user ID and password specified in the TRANSFER-ADMISSION and, if necessary, against the file password. If this check is negative, the file transfer is not executed and the user obtains a appropriate message.

For data protection reasons, this message gives no indication which of the parameters USER-IDENTIFICATION, ACCOUNT or PASSWORD or the file password has been violated.

If a receive file does not yet exist, it is created by openFT. Here, too, openFT uses the products named above to check the access rights (write) against the user ID and password. If this check is negative, the file transfer is not executed, and the same message as above is output.

openFT does not assign access protection attributes to new files. In particular, no file password is given to the system and the "RACF bit" is not set in the DSCB of openFT. If you want to give immediate protection against unauthorized access to a new file under z/OS, you are recommended to use the RACF function "generic profile", which is described below.

This RACF function can be used to assign common protection to a group of files with a similar name structure. For example, all the files of a specific user ID with a name which contains the string TRANS can be protected against access by other user IDs. This also applies to files created by openFT.

The file protection attributes of the send file are not transferred to the receive system and therefore cannot be adopted for a new receive file.

openFT does not set a modification time limit for the file.

4.9 File management

File management in openFT is possible both in the remote and in the local system.

If the partner system is a z/OS system, a number of special issues need to be observed, e.g. no local file management is available on z/OS. You will find details in the manual "openFT (z/OS) - Command Interface".

4.9.1 File management in the remote system

openFT offers the option of managing remote system files from the local system (file management). In the partner system, you can

- list the contents of directories,
- query file attributes, e.g. query the size of a send file,
- modify file attributes, e.g. rename files,
- delete files,
- create, rename and delete directories.

Partner systems, which support the file management function can also assume the initiative for such requests and access your local system accordingly from the remote system. In both cases, the system in which the initiative has been taken sends a description of the request to the partner system. The partner system executes the request according to its conventions.

The file management functions are performed via the appropriate protocols (openFT, FTAM or FTP). You can detect differences in the protocols between openFT, FTAM and FTP partner systems by changing the file attributes. Depending on the protocol, and what the partner system supports, you can modify the following attributes of a file.

Attribute	**FTAM partner**	**openFT partner**	**FTP partner**
File name (FILE-NAME/NEW-NAME)	X	X	X
Access rights (ACCESS-MODE)	X	X	
File availability (FILE-AVAILABILITY)	X		
Account for file storage costs (STORAGE-ACCOUNT)	X		
Legal qualification for using a file (LEGAL-QUALIFICATION)	X		
Future file size (FUTURE-FILE-SIZE)	X		

4.9.2 File management in the local system

When using the FTAM functionality, you have the option of assigning special FTAM attributes to files in the local system for communication with FTAM partners (see "Virtual filestore"). The functionality offered by this approach allows you to display and modify FTAM attributes of a file in the local system.

The FTAM attributes exist only in the virtual filestore and are primarily valid for file transfer and file management with FTAM partners. In the local system, the operating-system specific setting of the file attributes remains unaltered. This means that files and file attributes can still be modified using commands specific to the operating system. For example, a file can be deleted using a system-specific delete command although the corresponding setting of PERMITTED-ACTION prohibits deletion of the file for FTAM partners.

The following table shows the file management functions in the local system:

FTAM attribute	**display**[1]	**modify**
FILE-NAME *	X	
STORAGE-ACCOUNT	X	
Type of last file usage *	X	
Name of last user of file *	X	
Date and time of last change of file contents	X	
DATA-TYPE	X	X
CHARACTER-SET *	X	X
RECORD-FORMAT *	X	X
Maximum record length (RECORD-SIZE) *	X	X
File availability (FILE-AVAIAIBILITY) *	X	
Access rights (PERMITTED-ACTIONS) *	X	X
Current file size in bytes (CURRENT-FILE-SIZE) *	X	
Possible file size in bytes (FUTURE-FILE-SIZE)	X	
Legal qualifications (LEGAL-QUALIFICATION)	X	

[1]Only the FTAM attributes marked with * are displayed for local file management; all attributes are displayed for remote file management.

The following FTAM attributes are evaluated for file transfers using the openFT protocol and in part for the FTP protocol:

- Data type (DATA-TYPE)
- Record format (RECORD-FORMAT)
- Maximum record length (RECORD-SIZE)

If the format attributes specified in the file transfer request are not consistent with these FTAM attributes, the request is generally rejected. To avoid this, the FTAM attributes can be deleted in the local file without deleting the file itself.

However, these FTAM attributes are only set for file transfer requests using the FTAM protocol (not for requests via the openFT or FTP protocol).

4.10 Special points for file transfer with FTAM partners

The FTAM functionality allows you to execute file transfer on the basis of ISO protocol ISO8571. The sections below describe special points for “FTAM specialists” with respect to transfer and mapping of FTAM-specific file attributes for file transfer with FTAM partners.

The FTAM protocol is supported on BS2000, Unix and Windows systems.

4.10.1 Virtual filestore

Any system that is to enable file transfer using FTAM protocols must make its files available to partner systems in a format that is defined by standard (ISO8571). For this purpose a file's attributes are mapped from the real filestore onto a virtual filestore and vice versa. The virtual filestore thus has no effect on the attributes of the files in the local system, but has only the tasks of transporting file attributes to the remote FTAM system. In the sections below, the criteria for describing a file in the virtual filestore are introduced. The format of the virtual filestore is defined by the FTAM standard. Basically, a distinction is made between three different groups of file attributes:

Kernel group

describes the basic attributes of the files. These are specified when the file is created. They include the file name, information relating to the file structure and file contents, and details of agreed file access rights.

Storage group

covers the storage attributes of files. The storage attributes include the file size, the file availability, the date and time of the last read or write access, as well as identification of the user who initiated this in access.

Security group

defines the security attributes for access protection.

Attributes of the kernel group

The attributes in the kernel group are set when the file is created, and contain the basic information on a file:

file name

contains the file name.

permitted actions

define which actions can be performed for a certain file:

- read file (READ-FILE)
- insert data unit (INSERT-DATA-UNIT)
- replace (overwrite) file (REPLACE-FILE)
- extend file (EXTEND-FILE)
- erase data unit (ERASE-DATA-UNIT)
- read file attributes (READ-ATTRIBUTES)
- modify file attributes (CHANGE-ATTRIBUTES)
- delete file (DELETE-FILE)

The *permitted actions* also define the method that can be used to access structured files (see also the section “FTAM files”).

- forwards (TRAVERSAL)
- backwards (REVERSE TRAVERSAL)
- any (RANDOM)

contents type

Defines the data structure and the method that can be used to access the structured data.

Attributes of the storage group

The attributes of the storage group describe the filestore properties, for example who last accessed the file, the type of access, and when. Some of these properties are automatically modified when the file is read or modified. However, they cannot be modified directly using user commands. You can influence directly modifiable attributes with openFT.

Attribute [1]	**Definition**
storage account *	identifies who is responsible for the file storage costs
date and time of creation	indicates the date and time of creation
date and time of last modification	indicates the date and time of the last modification
date and time of last read access	indicates the date and time of the last read access
date and time of last attribute modification	indicates the date and time of the last attribute modification
identity of creator	identifies the user who created the file
identity of last modifier	identifies the user who last modified the file
identity of last reader	identifies the user who last read the file
identity of last attribute modifier	identifies the user who last modified the file attributes
file availability *	provides information on whether a file is available immediately, or whether it must first be obtained, e.g. from an archive
filesize	describes the storage capacity occupied in the actual filestore. A file can thus differ in size in systems that display file types in different ways. Some filestores assign a multiple of a basic unit, e.g. blocks, for file storage. *file size* thus specifies a value that does not correspond to the file size
future filesize *	describes the future file size, i.e. possible file size after processing. The initiator can modify the *future file size* value. As soon as the file reaches the specified file size, the responder can increase the value with or without a warning to the initiator. Alternatively, the responder can reject the modification of a value with an appropriate error message.

[1] Attributes marked with * can be modified directly.

Attributes of the security group

The FTAM virtual filestore concept provides a security group for access protection.

Attribute [1]	**Definition**
access control *	indicates the conditions governing access to files. For example, this may include passwords for various types of access (read, insert, replace, extend), or locks that are used to regulate simultaneous access to a file by different users.
legal qualifications *	specify the legal status of the file and its usage. At present, there is no accepted interpretation of this attribute, i.e. its interpretation depends on the particular partner.

[1] Attributes marked with * can be modified directly.

> The mapping of the file access rights and the mapping of the FTAM attributes to the real file system is described in the manual "openFT (BS2000) - Installation and Operation" and in the manual "openFT (Unix and Windows systems) - Installation and Operation", respectively.

4.10.2 Addressing via Application Entity Title (AET)

In the OSI world, communication partners are represented by application entities. An application entity is an addressable entity in Layer 7 of the OSI Reference Model (Application Layer). Such an application entity is the access point of an FTAM application, for example, via which an OSI-TP communication partner can connect to the FTAM application. In the OSI-TP standard, every application entity is assigned to an application entity title, via which the application entity can be addressed uniquely in the OSI network.

Two forms of AET are defined in the ISO Standard, the Directory Form (transparent form) and the Object Identifier Form (numeric form).

The FTAM functions of openFT (Unix systems) and openFT (Windows) support the Object Identifier Form of the AET. An AET comprises two parts:

- Application Process Title (APT)
- Application Entity Qualifier (AEQ).

When transmitting with the FTAM protocol, openFT sends a Nil Application Entity Title as a calling or called Application Entity Title by default. This behavior can be modified via operating parameter if desired.

On BS2000 systems, the transfer of Nil AETs can be set system-wide via an optional Rep.

The Nil AET is: 1.3.9999.1.7

4.10.2.1 Addressing FTAM partners with AET in Object Identifier Form

If a called AET is to differ from the "Nil Application Entity Title" then it must be specified in the partner list on instance identification.

The specification has the following syntax:

n1.n2[.n3] [.n10][..m]

n1.n2[.n3] [.n10]

specifies the *application process title*, between two and ten decimal numbers
separated by a period (.). The range and the meaning of the numbers are explained below.

[..m] specifies the *application entity qualifier*, range of *m* see below. The two periods are mandatory if a AEQ is specified.

Example

A FTAM partner on computer *daisy2* with APT=*1.0.56.881.4* and AEQ=*785* is to be entered in the partner list under the name *daisyftm*. To do this, enter the following command:

Unix systems and Windows systems:

```
ftaddptn daisyftm -pa=ftam://daisy2 -id=1.0.56.881.4..785
```

BS2000 systems:

```
ADD-FT-PARTNER PARTNER-NAME=DAISYFTM, -
               PARTNER-ADDRESS=FTAM://DAISY2, -
               IDENTIFICATION=1.0.56.881.4..785
```

Application Process Title (APT)

The APT used to identify the application. The APT should be unique worldwide in accordance with the OSI Standard. For this reason, it should be issued and registered by a Standardization Committee).

An APT in Object Identifier Form is consists of up to 10 components:

```
(component1,component2,...,component10)
```

The values for component1 to component10 are partially standardized. In this context, a symbolic name was assigned to several numbers. The range of values for component2 depends on the value of component1. The following table shows the symbolic names and the value ranges of the functions supported by FTAM:

component1	component2	component3 to component10
0: CCITT	0: RECOMMENDATION 1: QUESTION 2: ADMINISTRATION 3: NETWORK-OPERATOR (permissible values: 0 - 39)	Permissible values: 0 - 67 108 863

1: ISO	0: STANDARD 1: REGISTRATION-AUTHORITY 2: MEMBER-BODY 3: IDENTIFIED-ORGANIZATION (permissible values: 0 - 39)	Permissible values: 0 - 67 108 863
2:JOINT-ISO-CCITT	Permissible values: 0 - 67 108 863	Permissible values: 0 - 67 108 863

The APT which you specify need not be stipulated by a standardization committee, i.e. you may stipulate your own APT. It must satisfy the following two conditions:

- it must be unique throughout the network
- it must be made up of values that are permissible according to the table above

A remote partner that requests AETs must know this APT in order to set up a connection.

Application Entity Qualifier (AEQ)

The AEQ identifies an access point within an application. You can assign AEQs to the access points of an application only if you have assigned an APT to that application. It is assigned by the operator of the application.

The AEQ is a positive whole number between 0 and 67108863.

You must not use the same AEQ more than once within an application, i.e there must never be two access points with the same AEQ in one application. However, you do not have to assign all the access points in an application to an AEQ.

4.10.2.2 Extended support of the Application Entity Title on Unix and Windows systems

So far the sender was identified via the partner address. If the partner address of the initiator is changed from time to time, FTAM can have problems with the partner allocation and also with the recovery.

The Application Entity Title offers the option of implementing a sender verification which is independent of the partner address for FTAM as well. The sender verification of FTAM partners by means of the Application Entity Title can be specified via operation parameter settings.

Directory Form of the Application Entity Title

In addition to the numeric form there is a Directory Form (also known as transparent form) for Application Process Title and Application Entity Qualifier, respectively.

If an instance identification specified in a FTAM partner list entry does not correspond to the numeric form it is interpreted as Directory Form and the following applies:

- no blank at the beginning of a specification
- no blank at the beginning of a string that is attached to a numeric specification

Otherwise the Application Entity Title is rejected.

Specifications that contain a character which is neither a decimal number nor a dot will be sent and evaluated as ISO646 string in the Directory Form of the Application Entity Title.

The character string before the last occurrence of two sequential dots in the Directory Form is interpreted as Application Process Title.

The character string after the last occurrence of two sequential dots in the Directory Form is interpreted as Application Entity Qualifier. Exception: the identification ends with .. or ..#.

Examples

Specifications in the Directory Form are output in hexadecimal form as they would appear in the trace:

-id=1.3.5.6..#	APT = 0x312e332e352e36 *)
-id=9.3.4.5.2	is rejected as only 0, 1 or 2 would be permitted at the first digit
-id=1.2.3.4.5.6.7..8.9	APT = 0x312e322e332e342e352e362e37 AEQ =0x382e39

-id=1.2.3.4.5.6.7.8.9.10.11 is rejected

-id=%ip139.28.87.55	APT = 0x2569703133392e32382e38372e3535
-id=emil	APT = 0x656d696c
-id=1.3.5A	APT = 0x312e332e3541
"-id=1.3.5 A"	is rejected
"-id=Emil Huber"	APT = 0x456d696c204875626572
-id=Emil.Huber	APT = 0x456d696c2e4875626572
-id=Emil..Huber	APT =0x456d696c AEQ = 0x4875626572
-id=Emil..	APT = 0x456d696c *)
-id=Emil..#	APT = 0x456d696c *)

*) These specifications are not suitable for a sender check via Application Entity Title if the partner is openFT. I. e., leading zeros, and strings ending with . or .. must be avoided in numeric specifications, and - with transparent format .. or ..# at the end must not exist.

4.10.3 Synchronous transfer of multiple files with FTAM

It is possible to transfer successively multiple files with one *ncopy* command (Unix and Windows systems) or FTSCOPY command (BS2000 systems) via one transport connection and one FTAM connection. This works for send and receive requests and is controlled by the file name syntax.

The total length of the file list must match with the file name parameter of the ncopy/FTSCOPY command.

This function is also available at the program interface on Unix and Windows systems.

Specification of the send files

If the name of the send file does not start with two commas, everything will work as before.

If the name begins with two commas, the following applies

- Several names are derived from this name, which in turn are separated by two commas.

  Example 1

  `,,Letter1,,Document,,Booking`

  The files *Letter1*, *Document* and *Booking* are transferred.
- If the first subname ends with slash, point, or backslash in Windows, it is interpreted as a prefix for all subsequent subnames.

  Example 2

  `,,MyDirectory/,,Letter1,,Document,,Booking`

  The files *MyDirectory/Letter1*, *MyDirectory/Document* and *MyDirectory/Booking* are transferred.

Files with a comma at the end of the file name, or multiple commas within the name can not be transferred with the multiple option. Files whose names start with two commas can generally only be transferred asynchronously.

Specification of the receive file

For the single transfers, the name of the receive file is formed from the specified name and the name of the send file, whereby a prefix, if any, is not taken into account in the send file name.

Example

> If *target/* is specified as a receive file in Example 1 and Example 2, the receive files *target/Letter1*, *target /Document* and *target/Booking* are generated in both cases.

Logging and behavior in case of transfer errors

A separate logging record is written for each partial transfer request, and also follow-up processing (if specified) is started individually for each partial transfer request.

As a result of such a multiple file transfer, an OK message is received only if all transfers have been successful. Otherwise, the multiple transfer will be aborted after the first error and a corresponding message will be displayed even if some files of the multiple transfer have already been successfully transferred. Files whose transfer has not yet been started are marked as "canceled" in the logging.

5 Security functions

openFT provides the following security functions:

- FTAC functions
- Authentication
- Extended authentication check
- Encryption for file transfer and file management
- Protection mechanisms against data manipulation
- Encryption with FTPS

5.1 FTAC functions

FTAC provides the functions for controlling FT activities on a computer-specific and userspecific basis using admission sets and admission profiles:

- Security levels (0 to 100) are defined for a user ID in the admission set. The security levels have an influence on which file transfer functions this ID can use and which partner systems it can use them with.
- Admission profiles define a transfer admission which must be specified in file transfer requests instead of the LOGIN or LOGON authorization for the remote system. Access rights to a user ID are defined by restricting the use of parameters in file transfer requests.
- If FTAC is used, the FT administrator must assign security levels to the partner systems, see section “FTAC security levels for partner entries”.

> **WARNING!**
>
> Note that openFT-AC is only effective for connected products such as openFT. If other file transfer products without an openFT-AC connection are also being used, a more comprehensive and coordinated security concept would be advisable.

5.1.1 Admission sets

Admission sets implement the security settings on the basis of the basic functions of openFT. Administration of the admission sets is primarily the task of the FTAC administrator.

There are:

- The standard admission set that is valid throughout the system
- Individual admission sets for individual user IDs

Standard admission set

The settings made in the standard admission set apply to all user IDs. Consequently, only the FTAC administrator may modify the standard admission set.

Following installation, the standard admission set on BS2000 and z/OS systems is set to 0, i.e. no file transfers are possible. On Unix and Windows systems, the standard admission set is set to 100, i.e. file transfer is possible without restriction.

Individual admission sets

A maximum security level is specified in the admission set for each of the six basic functions (inbound send, inbound receive, inbound follow-up processing, inbound file management, outbound send, outbound receive).

Users may only modify their own admission sets whereas the FTAC administrator can modify the admission sets of all users. As a result, for these basic functions, there is a predefined value set by the FTAC administer (known as ADMIN LEVELS) and the predefined value set by the user in question (USER LEVELS). This results in the following possibilities for each basic function:

- If neither the user in question not the FTAC administrator have changed a setting then the setting in the standard admission set applies.
- If either the user or the FTAC administrator have changed the security level then this level applies. A user-specific security level must be the same as or lower than that in the standard admission set. If the security level is higher then a warning is output.
- If both of them have changed the setting then the lower of the two new settings applies.

Example

The *Valid level* column indicates the effect of the settings made by the user and the FTAC administrator as well as those in the standard admission set.

Basic function	User	FTAC administrator	Standard admission set	Valid level
Outbound Send	--	--	100	100
Outbound Receive	--	90	100	90
Inbound Send	50	--	90	50
Inbound Receive	50	10	90	10
Inbound Processing	50	10	50	10

Inbound File Management	10	20	50	10

-- means that no setting has been made

The user ID to which the admission set belongs can then use the basic function with all partner systems that have at most this security level, i.e., in the case of an openFT request (outbound and inbound), the admission is compared with the FTAC security level of the relevant partner, see also "FTAC security levels for partner entries".

Example

For the partners FT1, FT2 and FT3, the *Effect* column shows the effect of the settings in the administration set and the settings for the partner level.

Basic function	**Level in admission set**	**Partner level**			**Effect**		
		FT1	FT2	FT3	FT1	FT2	FT3
Outbound Send	100	100	90	10	+	+	+
Outbound Receive	90				-	+	+
Inbound Send	50				-	-	+
Inbound Receive	10				-	-	+
Inbound Processing	10				-	-	+
Inbound File Management	10				-	-	+

\+ Request is accepted
\- Request is rejected

5.1.2 Admission profiles

An admission profile is linked to a user ID, see section “Admission checking workflow” (Admission profile (FT profile)). Thus, it is usually the task of each user to administer his own admission profiles (creating, displaying, modifying, deleting).

An admission profile includes the following and other items:

- A transfer admission. This transfer admission must be unique. If a request is to work with the FT profile, this transfer admission must be specified. FTAC only permits access rights for this request which are defined in the FT profile. In order to uniquely assign the responsibility for request, it is recommended that a transfer admission be assigned to exactly one person in precisely one partner system.
- If necessary, specifications relating to transfer requests such as file name, file name prefix, follow-up processing commands or prefix and/or suffix for follow-up processing commands.
- If necessary, specification of the partner systems which may access this FT profile.
- Specifications relating to permitted file transfer functions and transfer direction, writing rules or encryption.
- If necessary, specification of whether and how long the FT profile is valid.
- Specifications indicating whether, and to what extent, the profile can ignore the set values entered in the admission set. Users can always ignore their own entries. Only privileged profiles can ignore the set values entered by the FTAC administrator, see “Privileged admission profiles”.

Example for a file name prefix

A file name prefix contains a part of the file or path name. The user of the profile can only navigate below this specified path name.

- C:\Users\Hugo\ as a file name prefix on a Windows system means that the user of this profile can only access directories and files below the path C:\Users\Hugo\. The same principle applies on a Unix system if, for example, /home/hugo/ is specified as a file name prefix.
- For example, if you specify PREFIX = USER. in BS2000 then an FT request in which FILE-NAME = HUGO has been specified will access the file USER.HUGO.
- On z/OS, for instance, a filename prefix is understood to be the "first-level qualifier" and where appropriate one or more further qualifiers, e.g. 'OPUSERS.HUGO.NEW.'.

This prevents anyone with this profile to navigate within locked directories or from using the preprocessing function. Note, however, that it is also possible to specify a remote preprocessing command as the file name prefix, in which case, only the parameters for that command would then need to be specified in the request.

Effects of an admission profile

The following table contains possible restrictions to the access rights in an FT profile in the left-hand column, and the entries for the file transfer request required for the partner system in the right-hand column. Some differences apply to a standard admission profile. See above.

Entry in the FT profile	Entry in the file transfer request
Transfer admission	The transfer admission addresses the admission profile. If the user ID and password are specified, it is only possible to address the standard admission profile of the user, if this has been defined.

Transfer direction restricted	The parameter specified must be the opposite of the entry in the FT profile. If the profile contains transfer direction “From Partner”, the remote system may only send data to the local system; with “To partner”, it is only possible to transfer files to the remote system. Therefore, only read access is permitted in the local system.
Partner systems specified	The request can only be issued by the partner systems entered in the profile.
File name specified	The file name must be omitted in the request. If it is a mandatory parameter in the file transfer product of the partner system, it must be assigned the value “*not-specified” (e.g. BS2000 systems).
Prefix for the file name specified	Only part of the file name which is not is present in the request. FTAC supplements this entry with the prefix defined in the profile to obtain the complete file name. The specification of absolute file names, or exiting a directory with “..” is prohibited by FTAC.
Processing prohibited	No processing may be requested for your processor.
Processing specified	No processing may be requested for your processor.
Prefix/suffix for followup processing specified	Only that part of the follow-up processing not defined in the profile may be specified in the request. FTAC supplements this entry to produce the complete follow-up processing command. If no follow-up processing is specified in the request, none is carried out.
Write mode restriction	The request is executed only if it complies with this write mode.
Force or forbid encryption	The request will only be carried out if it corresponds to the presets in the admission profile.

Privileged admission profiles

An admission profile is said to be privileged if the owner of the profile can override the (administrator's) entries for his admission set. Privileged admission profiles are only intended to be used in exceptional circumstances, e.g.:

- if a particular file needs to be transferred,
- if follow-up processing is not permitted or severely restricted,
- if a partner system with a higher security level is permitted to carry out file transfers with the user ID, but others with lower security levels are not.

In a privileged admission profile, users may only modify the transfer admission and reset the privileged status. This prevents the misuse of admission profiles that have previously been privileged.

Only the FTAC administrator is able to privilege an admission profile.

Notes on the standard admission profile

Unlike a normal profile, a standard admission profile has no FTAC transfer admission, because access is controlled implicitly using the user ID and password. On the other hand, this profile allows most of the normal parameters to be set, such as the permitted FT functions, a filename prefix or the write mode. You cannot set the expiry period, whether or not the profile is locked and whether the profile is private or public.

A standard admission profile must be set up explicitly and a maximum of one standard admission profile can be set up for each user ID.

5.1.3 The FTAC logging function

openFT-AC checks the access rights of every FT request which the protected system is involved in and logs the results. This information is stored in the so-called FTAC logging records. In addition to the normal logging data (timestamps etc.), FTAC logging records contain the following information that is of importance for the FTAC administrator:

- function of the FT request
- reason for any rejections of the request by FTAC
- transfer direction of the FT request
- name of the partner system with which the FT request was/is to be carried out
- LOGON authorization (USER-IDENTIFICATION) of the initiator of requests which were made in the local system (or *REMOTE for remote request initiators)
- name and privileging identifier of any admission profiles used
- the local file or library name (on BS2000 and z/OS)

FTAC only checks the admission for a request on the basis of the admission sets and admission profiles. openFT logs whether or not it can actually execute the request in the FT or ADM log records.

The display of FTAC logging records can not be turned off. However, the display can be restricted to requests rejected by FTAC or to modifying requests.

The FTAC administrator can obtain information about all the admission checks that FTAC has so far performed. This simplifies system review activities, for example.

Deleting logging records

The FT administrator and the FTAC administrator are the only users in the system who can not only view but also delete the FTAC logging records. The FT user can view only his own log records, he may not delete log records.

FTAC logging records can only be deleted from the oldest date up to a specified date. This ensures that there will be no gaps in the log file up to the most current record.

In theory, FTAC can write any number of logging records (“until the disk is full”). From time to time, the FTAC administrator should make a backup of existing logging records (either print out a hard copy or make a copy on tape or save a file in CSV format) and then delete these logging records from the log file. This ensures that the logging records will provide a continuous record over an extended period of time, as well as prevent the log file from getting too large. The FTAC administrator can change the current log file and retain older log records in offline log files.

5.2 Authentication

> The concept described in this section is available for openFT partners as of V8.1 running on Unix and Windows systems as well as for openFT partners as of V9.0 running on BS2000 and z/OS. This form of authentication is not available for FTAM partners and FTP partners because neither the FTP protocol nor the FTAM protocol that has been standardized by ISO provides a comparable functionality.

If data requiring an extremely high degree of security are to be transferred, it is important that the respective partner system undergo a reliable identity check (“authentication”) before the transfer. The two openFT instances that are engaged in a transfer must be able to mutually check each other using cryptographic means, to ensure that they are connected to the “correct” partner instance.

The openFT concept is based on the addressing of the openFT instances, using a networkwide, unique ID, and the exchange of partner-specific key information.

Please note that authentication is only possible for named partners!

5.2.1 Authentication usages

Basically, there are three distinct usages:

- Case 1:
  The local openFT instance checks the identity of the partner instance. This assumes that a current, public key of the partner instance was stored locally, see section “Keys of partner systems”.
  A configuration of this kind makes sense, for example, if a server’s files are to be accessed via openFT. It is important for the local openFT instance, that the received data come from a reliable source (the authenticated partner). In contrast, the source of an access attempt is unimportant to the server.
- Case 2:
  The partner instance checks the identity of the local openFT instance. This requires that a current, public key of the local openFT instance is stored in the partner instance (recoded in the case of Unix and Windows partners), see section “Local RSA key pairs” and section “Secure distributing of keys”.
  A configuration of this kind would be conceivable, for example, if partner systems in several branch offices were to be accessed from a central computer via openFT and the branch computers were only permitted to access the central computer (and, in fact, only the central computer).
- Case 3:
  Both of the openFT instances engaged in a transfer authenticate each other (combination of case 1 und case 2). This assumes that current, public keys were mutually exchanged and the partners are addressing each other using their instance IDs. In this way, it can be ensured that the data not only comes from a reliable source, but that it will also end up in reliable hands.

In the case of configuration errors that inhibit the authentication of one of the partners participated in the request no session is created which can execute the request. As a result, the problem can be identified not on the basis of the request status but of the partner status. The partner state (RAUTH or LAUTH) shows on which side the problem was recognized.

5.2.2 Instance identifications

The instance ID is a name up to 64 characters long, that must be unique throughout the network irrespective of case. It is particularly important when authentication is used.

During installation, the name of the local computer in the local network is defined by default as the instance ID. If it cannot be guaranteed that this name is unique in the network then the FT administrator must change the local instance ID via operation parameter settings.

Partner instance IDs

You store the instance identifications of partner systems in the properties of the partner system. Consequently, the partner system must be entered in the partner list, see section “Partner list”. With the aid of the partner systems’ instance IDs, openFT manages the resources assigned to those partners, such as request hold queues and cryptographic keys.

5.2.3 Local RSA key pairs

RSA keys are used for authentication as well as for the negotiation of the AES key with which the request description data and file contents are encrypted.

5.2.3.1 Key pair attributes

Each RSA key pair consists of a private and a public key. For BS2000 systems and z/OS there can be up to three key pair sets each consisting of three key pairs with lengths of 768, 1024, 2048. For Unix and Windows systems there are two additional key pair sets consisting of two key pairs each in the lengths 3072 and 4096. When a key pair set is created, new key pairs for each of these lengths are always created.

Public keys are stored under the following name:

SYSPKF.R<key reference>.L<key length>

The storage location is platform-dependent, see the relevant openFT "Installation and Operation" manual.

The key reference is a numerical designator for the version of the key pair. The public key files are text files that are created using the character code of the respective operating system, i.e. by default:

- BS2000 systems: Value of the system parameter HOSTCODE
- z/OS: IBM1047
- Unix systems: ISO8859-1
- Windows systems: CP1252

Private keys are internally administered by openFT.

> A key of length 2048 is used by default for encryption. The FT administrator can modify this setting via operating parameters.

Comments

For each key pair set, comments can be stored which are written in the first lines of the public key files when a key pair set is created. The comments could, for example, contain the contact information of the FT administrator on duty, the computer name, or similar information that is important for partners. The comments in the editable text file SYSPKF.COMMENT can only be a maximum of 78 characters long. When a key pair set is updated, any subsequently updated comments are taken over from this file into existing public key files.

5.2.3.2 Updating key files and replacing keys

The use of keys provides additional security. This security is also ensured in the long term by the following possibilities:

- The public key files for the existing key pair sets can be created again. This is necessary, for example, if a public key file had been accidentally deleted or manipulated in any other way.
- A key pair set can be replaced by a completely new key pair set. The most recent public key is identified by the key reference with the highest value in the file name. openFT supports a maximum of three key pair sets at the same time. However, multiple keys should only exist until the most recent public key has been made available to all partner systems.
- Key pair sets that are no longer needed can (and should!) be deleted immediately.

5.2.3.3 Imported keys

Keys consist of text files. It is therefore possible, in principle, to enter these files into the local system using the resources made available by the operating system (i.e. "manually") and copy them to the location where openFT expects to find the keys. However, this method is time-consuming and liable to errors and, in addition, special administrator rights would be needed on certain systems.

openFT therefore provides a number of built-in functions that can be used to import the following keys.

- Public keys of partner instances. These keys must have been generated by the partner's openFT instance.
- Private keys that were generated with an external tool (i.e. not via openFT). When importing a private key, openFT generates the associated public key. This key can be used in the same way as a key generated with openFT and distributed to partner systems.

Compared with the manual method, the import functions have the advantage that the keys (including newly created keys) are immediately present at the correct location in the local system.

Key formats

openFT supports key files in the following formats:

- PEM format (native PEM)

  The PEM-coded files must be present in EBCDIC format.
- PKCS#8 format encrypted without password phrase or after v1/v2 with password phrase (PEM-coded).
- PKCS#12 v1 format in the form of a binary file. The file is searched for a private key and any non-supported elements (e.g. certificates, CRLs) are ignored during the import. If the certificate is protected by a signature or hash then openFT does not perform a validity check. The validity of the file must be verified using other means. The first private key that is found in the file is imported. Any others are ignored.

The password phrase used for encryption must be specified in the password parameter when performing the import.

5.2.4 Keys of partner systems

The keys of partner systems can only be imported if the partner system is present as a named partner (see "Partner types") in the partner list because the partner name is used as reference.

If possible, the openFT import functions should be used for imports.

Properties of partner system keys

You can assign the following properties for the keys of partner systems:

- An expiration date, i.e. the key can no longer be used after this date has expired.
- Authentication level (1 or 2): If authentication level 2 is set, openFT performs additional internal checks. Level 2 is supported for all openFT partners as of Version 11.0B. Level 1 authentication attempts to this partner are rejected.

These properties can be defined for a specific partner or for all partners.

5.2.5 Secure distributing of keys

Distribution of public key files between the partner systems should take place using reliable means, for example by

- distributing them via cryptographically secure by e-mail
- distributing them on a CD (by courier or by registered mail).
- distributing them via a central, openFT file server, whose public key is in the partners' possession.

If you distribute public key files between partner systems with different operating systems, you must make sure that these files are recoded, e.g. from EBCDIC.DF04-1 (BS2000 system) to ISO 8859-1 (Unix system) or CP1252 (Windows system). If you transfer the key file as a text file then it is automatically recoded correctly.

5.3 Extended authentication check

Extended authentication checks (sender checking) are performed for partner systems that to not use authentication. Inbound requests are checked with the aid of the instance identification in order to ascertain whether the calling system has a valid entry in the partner list. openFT offers via sender checking the possibility of checking not only the processor name, but also the transport address.

The extended sender checking can be enabled globally for openFT partners via operating parameters or just for specific partners.

The global setting applies to all partners for which the "Partner attributes" automatic mechanism is active.

Extended sender verification is of no relevance for dynamic partners because these are always identified via the transport address.

If the authentication check returns a negative result, the request is rejected.

Special features of FTAM partners

On BS2000 systems the sender check for FTAM partners is always performed via the transport address. On Unix and Windows systems an FTAM partner can additionally be identified via the Application Entity Title which is independent of the partner‘s address. Operation parameter settings specify whether the sender check is performed via transport address, via AET or via both. For FTAM partner a partner-specific setting is not possible.

Special features of FTP partners

In the case of FTP partners, the sender check operates exclusively via the transport address. Consequently the "extended sender verification" attribute is ineffective for FTP partners and is also not displayed.

5.4 Encryption for file transfer and file management

openFT encrypts the request description data by default on file transfer. Optionally, it is also possible to encrypt file contents.

5.4.1 Encryption of the request description data

The partners involved in file transfer automatically negotiate encryption and use of the appropriate public key in the process of connection set-up.

If possible, openFT uses the RSA/AES procedure with a AES key length of 256 bits for encryption. In the case of connections with older partners, 128-bit RSA/AES or RSA/DES may also be used. In all cases, the most secure of the procedures that are supported by both partners is used.

In addition, a minimum ASE key can be set via operating parameters, i.e. only AES keys of the specified length or larger ones will be accepted. If the partner cannot fulfill this requirement then the request will be rejected.

openFT automatically encrypts the request description data if both partners support this functionality, there is an RSA key pair set in the local system and encryption has not been explicitly disabled. If you are the FT administrator you can set the key length required for the RSA key (RSA-PROPOSED) via the operating parameters. The default value after installation is 2048. Additionally you can specify a minimum RSA key length via the operating parameters.

If one of the acting instances has configured a minimum RSA key length, the openFT protocol assures that the negotiation of the AES key will be encrypted by an RSA key of that minimum length.

When one of the partners has either no valid RSA key or has switched off encryption while the other communication partner requires a minimum key length, no connection between the two partners will be possible.

5.4.2 Encryption of the file contents

It is only possible to encrypt the file contents (i.e. user data) if openFT-CR has been installed. This product is subject to export restrictions and is therefore not available in all countries. On BS2000 systems, the product CRYPT is used for encryption provided that it is installed and running. Otherwise openFT's internal encryption algorithms are used.

The encryption of user data is only available for:

- transfer requests with openFT partners
- outbound requests via the TLS protocol to a FTP server with Secure FTP support.

If one of the two systems is not capable of handling encrypted file transfers, the request is rejected.

Possible settings for file content encryption

openFT allows you to:

- Specifically request the encrypted transfer of your user data on outbound requests
- Force or prohibit the encryption of the user data via an admission profile in the case of inbound requests:
  - Encryption can be explicitly forced, e.g. for requests with particular security implications. Requests without encrypted user data are rejected.
  - Encryption can be explicitly prohibited, e.g. for requests with reduced security implications for which performance is important. Requests with encrypted user data are rejected.
- As FT administrator, you can use operating parameters to force data encryption for all inbound and/or outbound requests or to specify a minimum length of the AES key. If a minimum key length for the RSA and/or the AES key is specified, the behaviour is the same as described in the section “Encryption of the request description data”.

  The settings apply to file transfer requests via the openFT protocol as well as for administration requests. FTAM requests and inbound FTP requests are rejected because no encryption is permitted. File management requests are performed without encryption irrespective of the settings. In addition, the following applies:
  - If outbound encryption is activated then the file content is encrypted on outbound requests even if no encryption is demanded in the request itself. If the partner does not support encryption (e.g. because it is deactivated or because openFT-CR is not installed) then the request is rejected.
  - If an unencrypted inbound request is to be processed while inbound encryption is activated, then this request is rejected.

Please note that the effort required for encryption in the participating partner systems has a negative impact on performance. This means in particular that both the transfer time of files can become larger and the CPU usage can be increased.

5.4.3 Encryption for file management requests

openFT allows you to:

- Define for a „show“ commands that file attributes and directory list attributes are to be transferred encrypted.
- Force or prohibit encrypted transfer of file and directory list attributes via admission profiles.

The same encryption procedures are used as in the case of the encryption of request description data and file contents, i.e the statements to openFT-CR also apply to the file management encryption.

5.5 Protection mechanisms against data manipulation

During communications with openFT partners, openFT as of V8.1 implicitly checks the integrity of the transferred data. The request description data is always checked for integrity. The following applies depending on the options set in the request or via the operating parameters:

- In the case of requests with encryption, the integrity of the transferred file contents is also checked.
- In the case of requests without encryption, checking of the file contents can be explicitly requested in the order.

If an error is detected, restartable requests attempt a new transfer. Non-restartable requests are aborted.

In this way it is possible to detect and prevent malicious manipulations of the transferred data (e.g. in insecure public networks such as the Internet).

Errors on the physical transfer channels are identified and rectified by the communication system itself. No data integrity check at openFT level is required for this.

5.6 Encryption with FTPS

To ensure the security of FTP transfers, it is also possible to encrypt the transport connection using Transport Layer Security (TLS). This protocol is described in RFC4217 (Securing FTP with TLS) and is usually referred to as FTPS. The names *FTP Secure*, *FTP over TLS* or *FTP over SSL* (name of the predecessor protocol) are also commonly used.

A FTPS server makes its key and the certificate available to the openFT for encryption purposes. No mutual authentication is carried out.

An openFT client is able to exchange encrypted outbound user data with a FTPS server if openFT-CR is installed on the openFT side and the FTP server supports the TLS protocol. AES is used as the encryption method. If the openFT client requires encryption of the user data in the request, but the FTP server does not support the TLS protocol, the request is rejected.

If the openFT client does not require encryption of the user data, the request description data is encrypted if the FTP server accepts the TLS protocol, otherwise the request description data is transferred in unencrypted form.

6 Working with openFT

openFT provides users with the following interfaces:

- Command interface
- openFT Explorer for Unix and Windows systems
- Program interface
- openFT-Script interface
- ISPF panels for z/OS

6.1 Command interface

openFT provides a command interface on all platforms. This makes it possible to run openFT functions via procedures or scripts, i.e. operation is automated and tasks are performed in batch mode at certain times or when certain events occur.

The command interface provides return codes that make it possible to react automatically to errors, e.g. if a partner system cannot be reached.

There are two variants of the command interface:

- On BS2000 systems, the commands are available in SDF format, i.e. the standard format for BS2000.

  On z/OS systems, the commands are also available in SDF syntax. Except for the actual command names, the syntax is identical on BS2000 and z/OS systems: On z/OS systems, system constraints mean that only short names are possible (max. 8 characters), whereas in BS2000 the names can usually be very extensive.

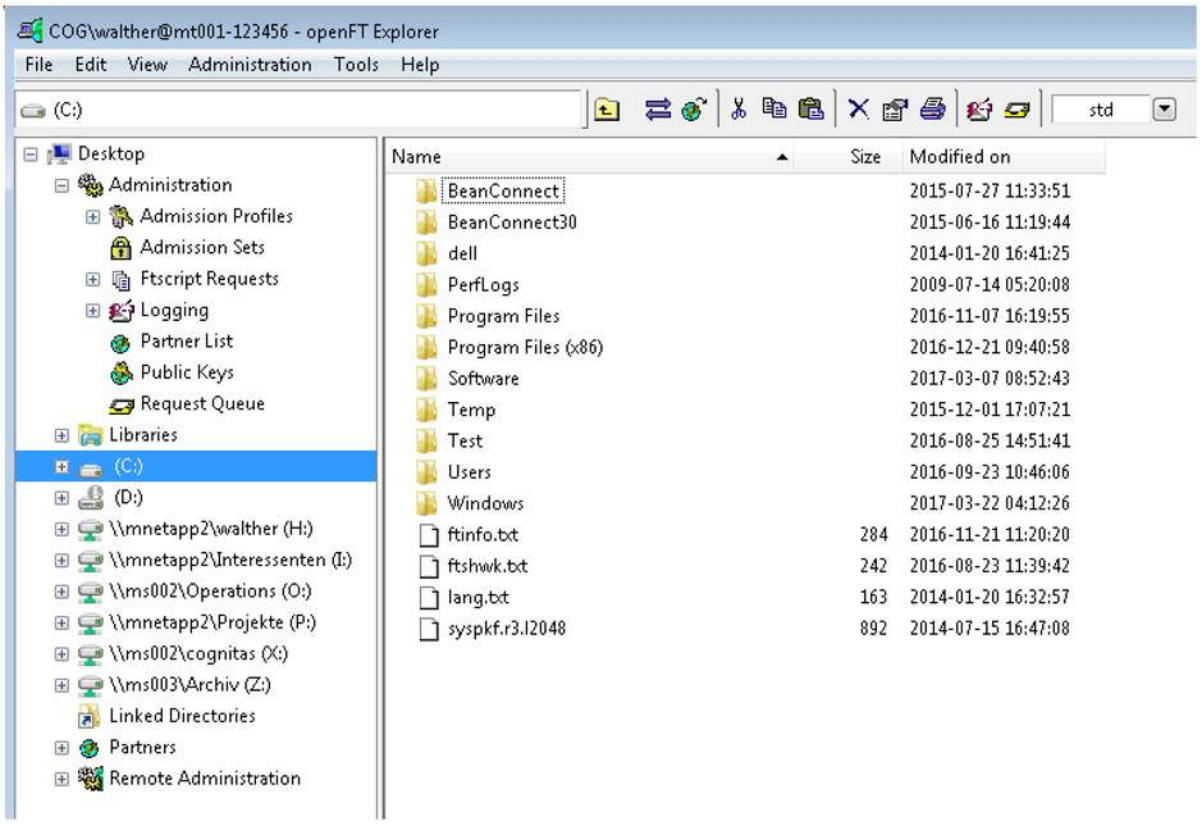
  There are only minor platform-specific differences in the functions.
- On Unix and Windows systems, the commands are available in the habitual format for Unix and Windows systems. The syntax is identical on the two systems and there are only minor differences between the functions. The syntax is also supported at the BS2000 POSIX interface.

On Unix systems, for each command there is also a corresponding man page, which you can call from the command line by using the Unix command *man* together with the name of a command as its argument (e.g. *man ft*).

For further details, see manual "openFT (BS2000) - Command Interface ", manual "openFT (z/OS) - Command Interface" and manual "openFT (Unix and Windows systems) - Command Interface".

6.2 openFT Explorer for Unix and Windows systems

openFT Explorer is a graphical user interface whose appearance and operation are similar to the Windows Explorer. The object directories appear in the left pane of the window, and the objects of the selected folder appear on the right, see the following example for Windows systems:



The openFT Explorer can be used to run all the functions present in the command interface, i.e. directory transfer, file transfer, file management, remote command execution and all administrative tasks. It is particularly suited to administrative tasks that do not constantly recur or are not easy to automate.

Alongside the openFT functions of the command interface, the openFT Explorer also provides a range of additional functions, such as:

- Configuration of Central Administration. To make this possible, the openFT-Explorer provides the integrated Configuration Editor.
- Remote administration
- Editing of text files using the integrated text editor
- Update file trees
- Simple file and directory transfer using copy & paste or drag & drop.
- User-specific partner folder: Users can create their own lists containing frequently used partners.

6.3 Program interface

The functions of openFT can be called via a platform-specific program interface. This makes it possible, for example, to integrate openFT functions in existing programs.

The program interface exists in the following variants:

- BS2000 systems: Assembler macros and Cobol calls.

  Both interfaces include file transfer, file management and full FTAC functionality. For further details, see manual "openFT (BS2000) - Program Interface".

- z/OS: Assembler macro

  The macro processes the openFT command string. This makes it possible to use the full function scope of the command interface. For further details, see manual "openFT (z/OS) - Command Interface".

- Unix and Windows systems: C and Java

  Both interfaces include file transfer, file management and the execution of remote commands (ftexec function). For further details, see manual "openFT (Unix and Windows systems) - Program Interface".

- Windows systems: OCX Control

  This interface makes it possible to execute synchronous data transfer requests via the interfaces (OLE/COM) of application programs. For further details, see manual "openFT (Unix and Windows systems) - Program Interface".

6.4 openFT-Script interface

openFT-Script provides you with a script language in XML notation. This comprises the openFT functions familiar from the command or C interface as well as offering additional context management and control functions.

Using openFT-Script, it is possible to combine multiple logically interdependent openFT requests within a single request (Ftscript). openFT-Script relieves the user of the tasks of monitoring consecutive openFT requests and provides a restart capability in the event of an interruption.

The XML statements in an openFT-Script request are stored in a text file. These files can be edited with a text editor or any desired XML tools. No compiler is required. The J2SE™ Runtime Environment 7.0 (JRE 7.0) or higher is required for execution.

An openFT-Script request is started using the ftscript command. In addition, the openFT-Script interface offers further commands for the administration of openFT-Script runs. openFT-Script requests can also be monitored and cancelled in the **Ftscript Requests** object directory in the openFT Explorer.

A detailed description of the XML interface can be found in the manual "openFT (Unix and Windows systems) - openFT-Script Interface".

6.5 ISPF panels for z/OS

On z/OS, openFT can be operated using so-called ISPF panels. For calling openFT functions via ISPF panels, the IBM program product "Interactive System Productivity Facility" (ISPF) must be installed on the system. ISPF provides a character-oriented user interface in the form of so-called "panels". The presentation corresponds to that of a 3270 terminal emulation, i.e. by default, 24 lines of 80 characters each. The example below illustrates a typical "start menu":

```
---  openFT - PRIMARY OPTION MENU  -----------------
OPTION  ===>
   1  ADMINISTRATION
   2  FILE TRANSFER REQUESTS
   3  EXECUTE REMOTE COMMANDS
   4  EXECUTE REMOTE FTADM COMMANDS
   5  ADMISSION SETS
   6  ADMISSION PROFILES
   INSTANCE IN USE ===>  STD
   COMMAND DISPLAY ===>  Y  (Y/N)
   ------------------------------------------------------
   |(C) 2020 Fujitsu Technology Solutions GmbH          |
   ------------------------------------------------------
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIIGHT   F12=RETRIEVE
```

The panels make use of a menu and question-driven method with immediate feedback on errors and available help functions. The panels make it possible to use practically all the openFT functions that are also available via the command interface.

When the FT user uses this menu interface, the corresponding commands are issued internally.

7 Administration

openFT administration comprises the following topics:

- Role concept for administration
- Local administration
- Remote administration via openFT Explorer
- Central administration

7.1 Role concept for administration

The administration of openFT is distributed over multiple roles so that the individual tasks can be performed by a number of different people.

openFT recognizes the following roles.

- FT administrator

  The openFT administrator monitors and controls FT operation, i.e. starting and shutting down openFT, setting operating parameters, administering partners, etc.

- FTAC administrator

  The FTAC administrator monitors FTAC operation, i.e. manages and backs up admission sets and admission profiles.

- ADM administrator

  The ADM administrator administers the remote administration server. As a result, an ADM administrator only exists for Unix and Windows systems, see also section “Remote administration”.

Who are the FT administrator and FTAC administrator?

The roles of FT and FTAC administrator possess special rights on all systems. The way in which the two roles are assigned to concrete user IDs depends on the platform:

- BS2000 systems:
  The FT administrator and FTAC administrator are defined via the BS2000 privileges FT-ADMINISTRATION and FTAC-ADMINISTRATION. By default, the two privileges are assigned to the BS2000 system administrator ID TSOS. If SECOS is used on the BS2000 system then the two privileges can also be passed on to another ID.

  For further details, see manual "openFT (BS2000) - Installation and Operation".

- z/OS systems:
  The user IDs for the FT and FTAC administrator are defined in a configuration file known as the FT parameter library. Only the system administrator may edit this configuration file. On z/OS systems, the roles of FT and FTAC administrator can both be assigned to multiple IDs.

  For further details, see manual "openFT (z/OS) - Installation and Operation".

- Unix and Windows systems:
  On Unix and Windows systems, FT administration rights are linked to the system administration rights, i.e.
  - In multi-user operation on Unix systems, all IDs with root permission (UID=0) possess FT administration rights.
  - On Unix systems with single-user operation, only the ID under which openFT is running possesses FT administration rights.
  - On Windows systems, only the Administrator account is an FT administrator by default if User Account Control (UAC) is activated. Other users can, however, run openFT applications such as the openFT Explorer with administrator permissions if they grant the relevant permission in the dialog box displayed by the operating system when the program is started.

  Following a new installation, the FT and FTAC administrators are identical. This means that all users who possess FT administration rights on the system are also FTAC administrators.

The FTAC administrator is identified by the fact that the corresponding privilege is defined in his or her admission set. If the roles of FT and FTAC administrator are to be separated, the FT administrator has to privilege the admission set of the (future) FTAC administrator. This is useful, for example, if someone other than the system administrator is responsible for data security.
See the manual "openFT (Unix and Windows systems) - Installation and Operation" for more details.

7.2 Local administration

The tasks of the FT administrator include:

- Starting and stopping openFT
- Administering openFT operating parameters
- Administering RSA keys
- Monitoring and controlling openFT operation
- Administering partners
- Diagnostics and error correction

The tasks of the FTAC administrator include:

- Administering admission sets and admission profiles

7.2.1 Starting and stopping openFT

The tasks differ on mainframe systems and on Unix/Windows systems:

- On mainframe systems, it is first necessary to start the openFT subsystem (BS2000) or load openFT (z/OS). Although the file transfer functions are then available, only synchronous outbound requests are executed. Other local requests are stored in the request queue. In order to execute asynchronous requests (outbound and inbound), the FT administrator must explicitly start the openFT server process via a command.

  The FT administrator can explicitly stop the openFT server process via a command again and restart it.
- On Unix and Windows systems, openFT is normally automatically started and ended together with the system. The openFT administrator can stop the asynchronous openFT server (and start it again) while the system is running. If the asynchronous openFT server has been stopped then only synchronous outbound requests are executed. Other local requests are stored in the request queue. This function can be used, for example, to suspend inbound operation during maintenance operations or for reasons of security.

7.2.2 Administering openFT operating parameters

openFT possesses a number of settings that can only be configured and, if necessary, modified, by the FT administrator. These are also referred to as the openFT operating parameters. Following installation, the parameters are set to default values that are based on experience and are usually adequate for normal operation.

The FT administrator therefore only needs to modify these values if this is necessary for the system in question. The settings listed below are present on all platforms:

- Settings for the local openFT system:
  - Identification and name of the local system
  - Addresses for the individual protocols (openFT, FTP, FTAM)
- Performance-related settings
  - Maximum values for operation, e.g. for the number of parallel connections, the number of requests in the request queue or the lifetime of requests
  - Scope of recording of monitoring values, see "Controlling the FT monitoring function"
- Security settings:
  - Default value for partner security levels
  - User data encryption
  - Length of the employed RSA key
  - Minimum RSA key length
  - Minimum AES key length
  - Mode for sender checks
- Logging settings:
  - Extent of logging
  - Automatic deletion of log records
  - Enabling/disabling of traces
  - Switch-over of log file and trace file
  - Scope of traps (console traps, ADM traps and possibly also SNMP traps)

Some settings are only available on certain platforms:

- Setting the variant of the employed code table (Unix/Windows systems and z/OS)
- Defining the settings for the remote administration server (Unix systems and Windows systems)
- Defining the use of TNS and CMX (Unix systems and Windows systems)
- Activating and deactivating the Application Entity Title (AET) for FTAM operation (BS2000 systems, Unix systems and Windows systems)

For further details, refer to the corresponding openFT "Installation and Operation" manual.

7.2.3 Administering RSA keys

The administration of the RSA key pair sets is the responsibility of the FT administrator and consists of the following subtasks:

- Creation of RSA key pair sets for the local openFT instance
- Output of all keys in the local system
- Updating of the public keys
- Deletion of local RSA key pairs
- Modification of RSA key attributes
- Importing of RSA keys

7.2.4 Monitoring and controlling openFT operation

The monitoring and control of openFT operation primarily consists of the following subtasks:

- Administering requests
- Administering FT logging
- Administration via SMNP
- Controlling the FT monitoring function
- Evaluating console messages
- Evaluating ADM traps

7.2.4.1 Administering requests

All asynchronous outbound requests and all inbound requests are stored in the request queue. The FT administrator can obtain information about all the asynchronous requests that have not yet been completed on your computer. This also includes the right to query information about the requests for all users.

The FT administrator can

- modify the sequence in which requests are processed at the local system. This may be necessary, for example, if an urgent transfer request has to be completed immediately at times of high load (request queue full).
- delete asynchronous requests for the local system. This may be necessary, for example, if a request has been issued to a partner whose address has changed and the users who issued the request cannot be reached. The FT administrator must of course inform the users.

7.2.4.2 Administering FT logging

openFT can record the results of all file transfer requests, irrespective of whether the initiative is in the local or the remote system (outbound and inbound requests, respectively). The information on each successfully completed or aborted request is recorded in a so called "FT logging record". This represents a complete, uninterrupted documentary record of FT operation over a prolonged period of time.

If FTAC is used then FTAC logging is always also performed. The FTAC functionality is always installed and activated on Unix and Windows systems. FTAC is optional on mainframe systems.

If central administration is used, ADM logging for central administration is also available.

The FT administrator has the following tasks

- Setting the extent of logging via the operating parameters

  For example, logging can be restricted to only those requests for which an error has occurred. FT logging can be completely disabled. However, this is not recommended if uninterrupted logging is desired.
- Switching the logging file (offline logging)
- Backing up and, if necessary, deleting the log records
- Using the log records for diagnostic purposes, see section “Diagnostics and error correction”.

Changing the log file and administering offline log files

The FT administrator can change the log file via setting operating parameters. This closes the current log file which is nevertheless retained as an offline log file. For the following log records, a new log file is created with the current date in the suffix. The log file can be changed several times and therefore multiple offline log files can be managed.

This change-over has the following benefits:

- Faster access to logging information due to smaller log files.
- Improved administration of log records through regular change-overs and back-ups of the offline log files.
- Possibility of performing extensive searches in the offline logging information without affecting ongoing openFT operation.

Saving and deleting log records

Depending on the volume of requests, the FT administrator should back up the log records from the current log file or from the offline log file(s) at regular intervals, for example as a file in CSV format, and then delete these log records or offline log file(s). In this way, the log records remain available to provide uninterrupted documentation over an extended period, while simultaneously ensuring that storage space is not taken up unnecessarily.
On BS2000 and z/OS systems, it should be noted that deleting log records does not reduce the assigned file size of the current log file but simply that space in the file that is no longer required is freed up.

Automatic deletion of log records

The FT administrator can set the intervals for the automatic deletion of log records. This setting deletes log records as of a defined minimum age at regular intervals and at a specified time.
This automatic delete function is only active if openFT is started. If openFT is not started at a scheduled delete time then the delete operation is not performed on the next start-up.
Following installation, the automatic deletion of log records is disabled. The FT administrator should only enable this function if the uninterrupted recording of log records is not required.

7.2.4.3 Administration via SMNP

SNMP stands for Simple Network Management Protocol and was developed as the protocol for network management services in TCP/IP networks. openFT permits you to centrally monitor and administer one or more openFT systems from one central management station using graphical interfaces.

To support automatic monitoring, some events which are not direct responses to user input are reported by openFT via a console message. Console messages can also be used to generate SNMP traps for automatic FT monitoring using SNMP.

If the file transfer subagent on BS2000 systems is used then openFT itself can also generate SNMP traps (without having to use console messages).

Requirements

SNMP-based openFT management requires the presence of a number of products and components. Consequently, the following items must be additionally installed or activated on the various platforms.

- BS2000 systems: The products SNMP Management >= V6.0, SNMP-Basic-Agent BS2000 V6.0 (SBA-BS2) and SNMP-Standard-Collection BS2000 V6.0 (SSC-BS2) must be installed.
- Windows systems: Microsoft's SNMP service must be present. The openFT subagent is present in openFT and is set up during openFT installation.

Detailed information can be found in the respective user manuals.

Functional range of the openFT subagent

The file transfer subagent is used to:

- start and stop openFT (BS2000)
- acquire system parameter information
- change the public key for encryption
- output statistics
- diagnostic control
- output partner information

Detailed information can be found in the respective openFT manuals "Installation and Operation".

7.2.4.4 Controlling the FT monitoring function

openFT provides the option of monitoring and displaying a range of characteristic data for openFT operation. The data falls into three categories:

- Throughput, e.g. total network throughput caused by openFT
- Duration, e.g. processing time for asynchronous jobs
- State, e.g. number of requests currently queued

The FT administrator can enable monitoring data acquisition in order to gain an idea of the type of load. However, monitoring data acquisition should not be permanently enabled since this impairs performance.

The following options are available for the FT administrator

- Activating and deactivating monitoring
- Selective monitoring based on the partner type
- Selective monitoring based on the request type

Once the settings have been selected, they are retained until the FT administrator explicitly changes them. This means that they also remain unchanged after the computer has been rebooted.

Showing monitoring data

If monitoring is activated the monitoring data can be called up on the local system or from a remote system.

In principle, all users can read the monitoring data. However, in practice, it is the task of the FT administrator to analyze the data since he or she is the only person who has an overview of the entire system and is able to modify the operating parameters.

The monitoring values are output as a table.

On Unix and Windows systems, the monitoring data can also be displayed via the graphical user interface provided by the openFT Monitor. The openFT Monitor possesses "remote" capability, i.e. it can also display the monitoring values for remote systems. In the case of mainframe systems such as BS2000 or z/OS, this capability can be used as follows:

1. On the mainframe system, the FT administrator sets up a special admission profile that is specified when the openFT Monitor is called and causes only the monitoring values to be read and transferred. The admission profile uses the keyword *FTMONITOR as preprocessing command. HOSTMONITOR, for example, can be specified as the transfer admission.
2. On the Unix or Windows system, you specify the mainframe system with the transfer admission HOSTMONITOR when calling the FT Monitor. The FT Monitor outputs a graphical display of the mainframe monitoring values.

On Windows systems, it is also possible to display the monitoring data via the Windows Performance Monitor.

7.2.4.5 Evaluating console messages

To support automatic monitoring, some events which are not direct responses to user input are reported by openFT via a console message.

Typical events may take the form of status changes in openFT (Start/Stop) and partners or failed requests.

The FT administrator can specify the events that are to be reported by means of the operating parameters (console traps).

7.2.4.6 Evaluating ADM traps

ADM traps are short messages that openFT sends to the ADM trap server if certain events occur during operation of openFT. Such events may include errored FT requests, status changes or the unavailability of partners, for instance.

The FT administrator can specify the events that are to be reported by means of the operating parameters (ADM traps).

See section “ADM traps” for more details.

7.2.4.7 Other administration functions

The following administration functions are not present on all platforms.

Job variables on BS2000 systems

An openFT instance can be monitored by an automatically populated MONJV. The job variable is created the first time at the start of openFT and used thereafter.

Job log on z/OS

Beside the log file the openFT job log also contains information which may be useful for the FT administrator. Some messages are output only to the openFT job log; often, however, the chronological order of the messages contained in the job log is useful in the diagnosis of errors during FT operation.

7.2.5 Administering partners

The FT administrator can determine the partners with which file transfer and file management are possible and what security settings apply to which partners. The FT administrator is responsible for maintaining the partner list, i. e. he or she can

- Enter partners in the partner list

  When doing this, the FT administrator can assign the partner individual properties such as, for example:

  - Security level
  - Mandatory requirement for authentication
  - Priority level
  - Enabling/disabling of the trace for this partner
  - Permit/prohibit parallel processing of requests involving this partner
  - Switching on/off the restart function (recovery) for outbound requests
  - Status settings such as, for example, automatic deactivation of multiple failed connection attempts
- Modify the properties of partners in the partner list if necessary
- Delete partners from the partner list
- Allow or lock dynamic partners

  In this way, the FT administrator can, for example, prevent unknown partners from the Internet from accessing the local system via openFT.
- Export the partner list

  The FT administrator can use a command to output the entries in the partner list and export them to a file, for example in order to back up the entries or use them in other systems. In this command, it is possible to specify the platform for which the commands are generated, i.e. it is possible to export a BS2000 to Unix or Windows, for example. When the export is performed, the entries are converted into the corresponding commands which then simply have to be imported into the required system.

For further details on the partner list and partner types, see chapter “Partner concept”.

7.2.6 Diagnostics and error correction

The FT administrator possesses two important tools for diagnostic purposes:

- Log records

  If a serious problem occurs, the FT administrator should first try to obtain the associated FT, and possibly also FTAC, log records. The log records contain a return code that can provide initial information on the cause of a problem. A log record also contains all the important parameters of an FT request.

- openFT traces

  If the log records are not sufficient to permit a diagnosis then the FT administrator can enable an FT trace. The trace can be enabled globally or be more restricted, right through to traces that apply to a specific partner only.

  The FT trace files can then be prepared and analyzed using an openFT tool (*fttrace*).

  On Unix and Windows systems, the FT administrator can view the trace files directly with the openFT Explorer.

- Traces for lower protocol layers

  On Unix and Windows systems the FT administrator can enable traces for the lower protocol layers. This information is written to the openFT trace.

7.2.7 Administering admission sets and admission profiles

The administration of admission sets and admission profiles is the task of the FTAC administrator. This task involves the following activities:

- Defining the standard admission set
- Defining and maintaining individual admission sets
- Administering admission profiles
- Transferring the FTAC environment

7.2.7.1 Defining the standard admission set

The FTAC administrator must first determine an average protection level for the user IDs in his system and use this information to modify the standard admission set. In the standard admission set, the settings are made for the "average" FTAC user in the system. This provides adequate protection for most users. These specifications are valid for all user IDs which do not have their own admission set. Furthermore, in each admission set, the entry *STD can be used in different places to refer to the standard admission set. This has the advantage of automatically incorporating any modification of the standard admission set into these admission sets.

The FTAC administrator can set individual values for user IDs whose protection requirements deviate from the average.

7.2.7.2 Defining and maintaining individual admission sets

The FTAC administrator can define an individual admission set for each user ID. This can be valuable, for example, if the standard admission set is highly restrictive due to security considerations, e.g. no inbound access permitted (Inbound Send and Receive set to 0).

If, despite this, inbound transfers are to be permitted for an ID (i.e. initiated externally) then the FTAC administrator defines an individual admission set for this ID by increasing the security level for Inbound Send/Receive. The user may then also need to increase his or her own setting since this may still contain the original, more restrictive value from the standard admission set.

7.2.7.3 Administering admission profiles

The FTAC administrator has the option of modifying foreign admission profiles:

- He can create admission profiles for external user IDs. No particular restrictions apply in this case.
- He can view them. The transfer admission of an admission profile is not output. This means that the FTAC administrator does not have access rights to the files of foreign user IDs.
- He can delete them. This is the most radical of all options which should only be used in extreme cases and with good reason and upon consultation with the owner of the profile.
- He can privilege them, or conversely revoke privileges, see "Administering admission profiles".
- He can also modify them. If the FTAC administrator neither possesses the necessary system administrator rights nor enters the full LOGON/login authorization of the owner of the profile, the admission profile is locked until the owner of the profile acknowledges this change by setting the transfer admission to "valid" again.

Creating admission profiles for external user IDs

There are a number of different ways in which the FTAC administrator can create an admission profile for an external user ID:

- If the FTAC administrator possesses the necessary administration permissions (see section “Role concept for administration”) then he or she can set up admission profiles for other IDs without restrictions even without knowing the current user password. In these profiles, the FTAC administrator can specify a transfer admission which can be used in FT requests immediately after being created. You should note that FTAC administrators with these administration permissions can set up corresponding admission profiles that give them access to the files of all user IDs and may therefore be able to bypass security regulations!

  > ℹ The necessary administration permissions depend on the platform:
  >
  > - BS2000 systems: TSOS privilege
  > - z/OS systems: SU privilege
  > - Unix and Windows systems: FTAC administrator permissions

- If the FTAC administrator does **not** possess the necessary administration permissions, there are two possibilities:
  - He can enter the complete LOGON/login authorization (i.e. user ID, password and possibly also account number). He can then also specify a transfer admission. In this way, he creates a valid admission profile, i.e. this profile can be used immediately in file transfer and file management requests.
    However, the user's password is stored as a fixed element in this type of admission profile. If the user wants to change the password then it is also necessary to modify the admission profile.
  - He simply specifies the user ID (without password and, if applicable, account number). The profile is then created without a transfer admission, which must then subsequently be assigned by the user.

Privileging admission profiles

The procedure to follow when privileging an admission profile is simple:

1. The user creates an admission profile for the planned task
2. The FTAC administrator views the admission profile to determine if the profile presents a threat to data security.
3. If the profile will not endanger security, the FTAC administrator privileges it.

7.2.7.4 Transfer FTAC environment - the environment functions

The FTAC administrator can have admission profiles and sets written (i.e. "exported") to a file and thus back up all admission profiles and sets that exist on the computer.

In addition, this function is useful when a user migrates from one computer to another. In this case, the FTAC administrator first backs up the existing FTAC environment to a file and then re-installs this on another computer. The FTAC user can then continue to work in the same FTAC environment as before, i.e. with the same admission profiles and the same admission set.

Any existing privileges must be explicitly set up again on the new computer, and the admission profiles must be explicitly released if the FTAC administrator does not possess the necessary administration privileges.

On the other hand, if the FTAC administrator has the necessary administration privileges, he/she can specify on importing whether the profiles will be imported with unmodified attributes or not.

The FTAC administrator can also selectively back the FTAC environment by using corresponding parameter specifications and then restore them when needed:

- admission profiles and admission sets of one or more users (up to 100)
- all admission profiles and admission sets on a given computer
- only admission sets, no admission profiles
- only admission profiles, no admission sets

The FTAC administrator can view the contents of a backup file.

7.3 Remote administration via openFT Explorer

Some of the administrative activities that are available on the local system can also be performed on remote openFT systems by means of the openFT Explorer. This makes it very easy to administer remote openFT systems irrespective of the platform on which they are located. The function scope comprises the following local administration capabilities:

- Administering openFT operating parameters

  You can therefore set all the operating parameters for the remote system, see section “Administering openFT operating parameters”.

- Starting the openFT Monitor

  You can start the openFT Monitor on the remote system, see section “Controlling the FT monitoring function”. For this to be possible, openFT monitoring must be enabled, e.g. by means of the operating parameters via Remote Administration

In the openFT Explorer, these functions are available for all partners that are defined in the **Partners** node. To do this, you must save the transfer admission for the remote system. If you want to make use of the full range of functions, the specified transfer admission must possess the FT administration permission for the remote system.

7.4 Central administration

Central administration in openFT covers the functions **remote administration** and **ADM traps**. On Unix and Windows systems, openFT provides full support for both functions.

These functions offer considerable benefits if you want to administer and monitor a large number of openFT instances. These benefits include:

- Simple configuration

  The configuration data is maintained centrally on the **remote administration server**, which means that it only exists once. The creation of roles in the form of **remote administrators** and the grouping of several instances make it possible to implement even complex configurations simply and in a clearly structured way. Subsequent changes are simple to incorporate and thus make the configuration easy to maintain.

  The remote administration server runs on either a Unix or a Windows system.

- Simplified authentication procedure

  If you wish to use authentication for reasons of security, it is only necessary to distribute a few keys:

  - For the direction to the remote administration server, the keys of computers from which administration is to be performed must be stored on the remote administration server.
  - For the direction from the remote administration server to the instances to be administered, it is only necessary to store the public key of the remote administration server on the openFT instances to be administered.

- High performance

  The remote administration interface allows far longer command sequences than in openFT up to V10.0.

  In addition, it is possible to configure the remote administration server in such a way that it is available exclusively for remote administration. In this case, there is no dependency on normal FT operation and hence no mutual impact.

- Simple administration

  Remote administrators only need one (central) transfer admission. Without central administration, remote administrators have to remember the access data for each openFT instance to be administered.

- Central logging of important events

  ADM traps can be generated if certain events occur on openFT instances. These are sent to the (central) ADM trap server and stored permanently there. This allows remote administrators to evaluate important events at a later time and for specific instances.

- Compatible integration of earlier openFT versions

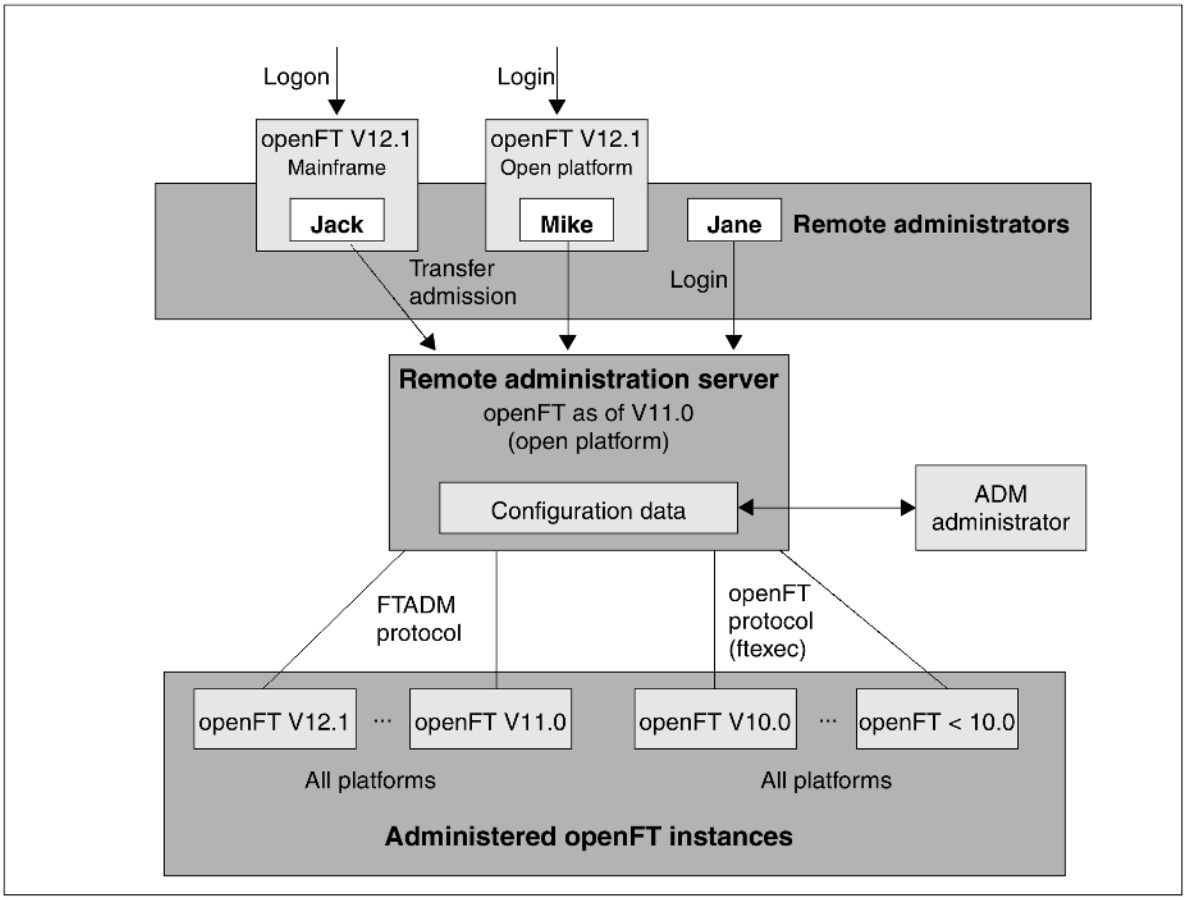
  Instances running versions of openFT as of V8.0 can simply be added to the configuration and administered in the same way as instances as of V11.0. All the administration functions offered by the corresponding openFT version can be used.

7.4.1 Remote administration

openFT allows you to set up a remote administration server via which you can administer your openFT instances on the various platforms. You can choose to use any openFT instance as an administration workstation.

7.4.1.1 The remote administration concept

The figure below shows the remote administration components and the most important configuration options on the basis of a deployment scenario.



Remote administration components

Remote administration comprises the following components:

Remote administration server

Central remote administration component. This runs on a Unix or Windows system with openFT as of V11.0 and contains all configuration data for remote administration.

Multiple remote administration servers can be defined in a complete configuration. See "Configuration with multiple remote administration servers".

ADM administrator

Person who administers the remote administration server. This person creates the configuration data for remote administration in which, for instance, the remote administrators and the administered openFT instances are defined. The ADM administrator is the only person permitted to change the configuration data.

The ADM administrator can use the graphical interface of the Configuration Editor for creating and editing easily the configuration data.

Remote administrator

Role configured on the remote administration server and which grants permission to execute certain administration functions on certain openFT instances. A remote administrator can

- Log in directly at the remote administration server (single sign-on)
- log in to a different openFT instance (as of V11.0) and access the remote administration server using an FTAC transfer admission.
  The openFT instance can be running either on a mainframe (BS2000/OSD, z/OS) or on a Unix or Windows system. The FTADM protocol is used for communication.

Several remote administrators can be configured with different permissions.

Administered openFT instance

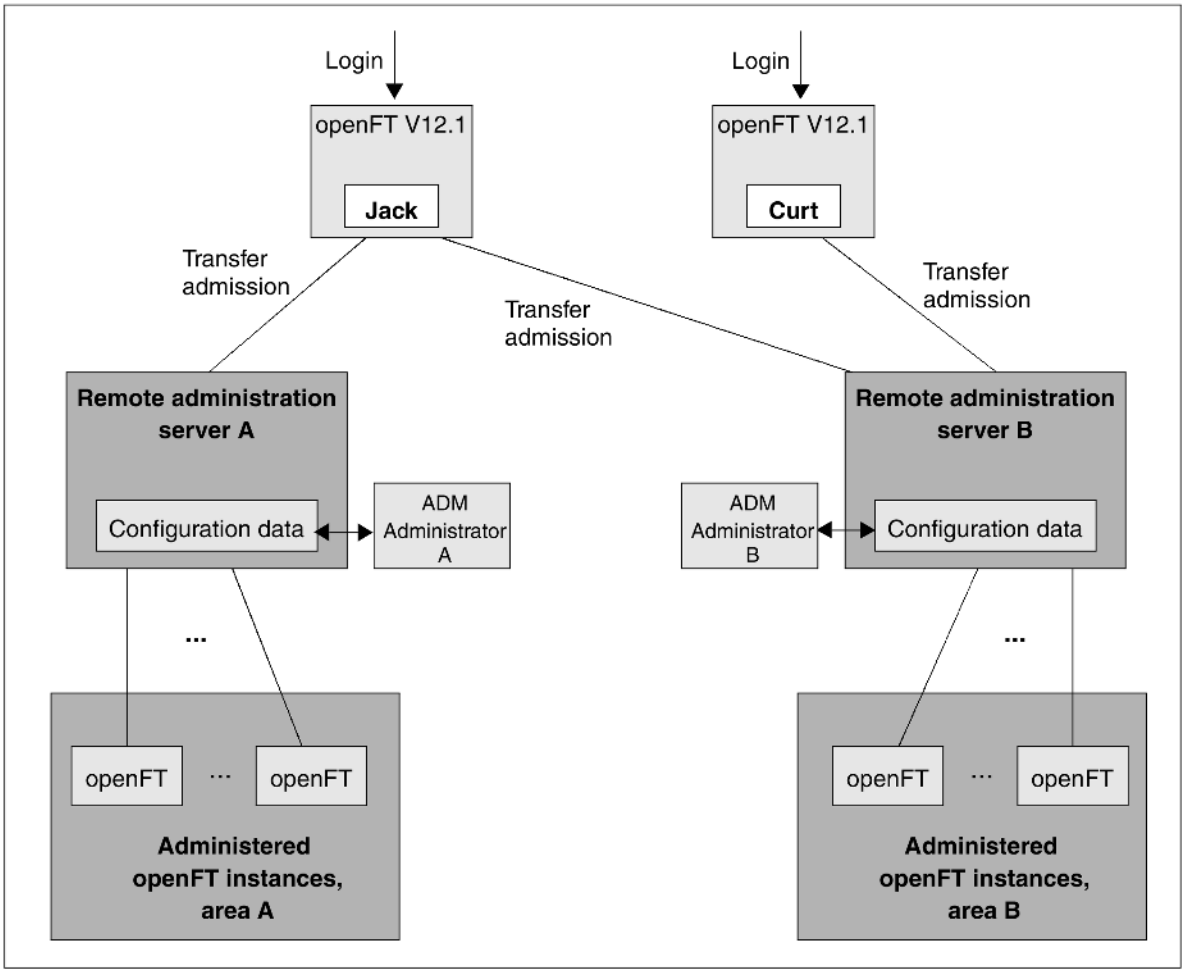
openFT instance that is able to be administered by remote administrators during live operation. Access is via an admission profile. The following applies, depending on the openFT version of the openFT instance:

- In the case of openFT instances as of V11.0, the FTADM protocol is used, and the full range of remote administration functions can be utilized.
- In the case of openFT instances from V8.0 through V10.0, administration is carried out using the openFT protocol and the command *ftexec*. The range of functions available depends on the openFT version of the instance being administered.

7.4.1.2 Configuration with multiple remote administration servers

Complex configurations can also be defined in which remote administrators access multiple remote administration servers.

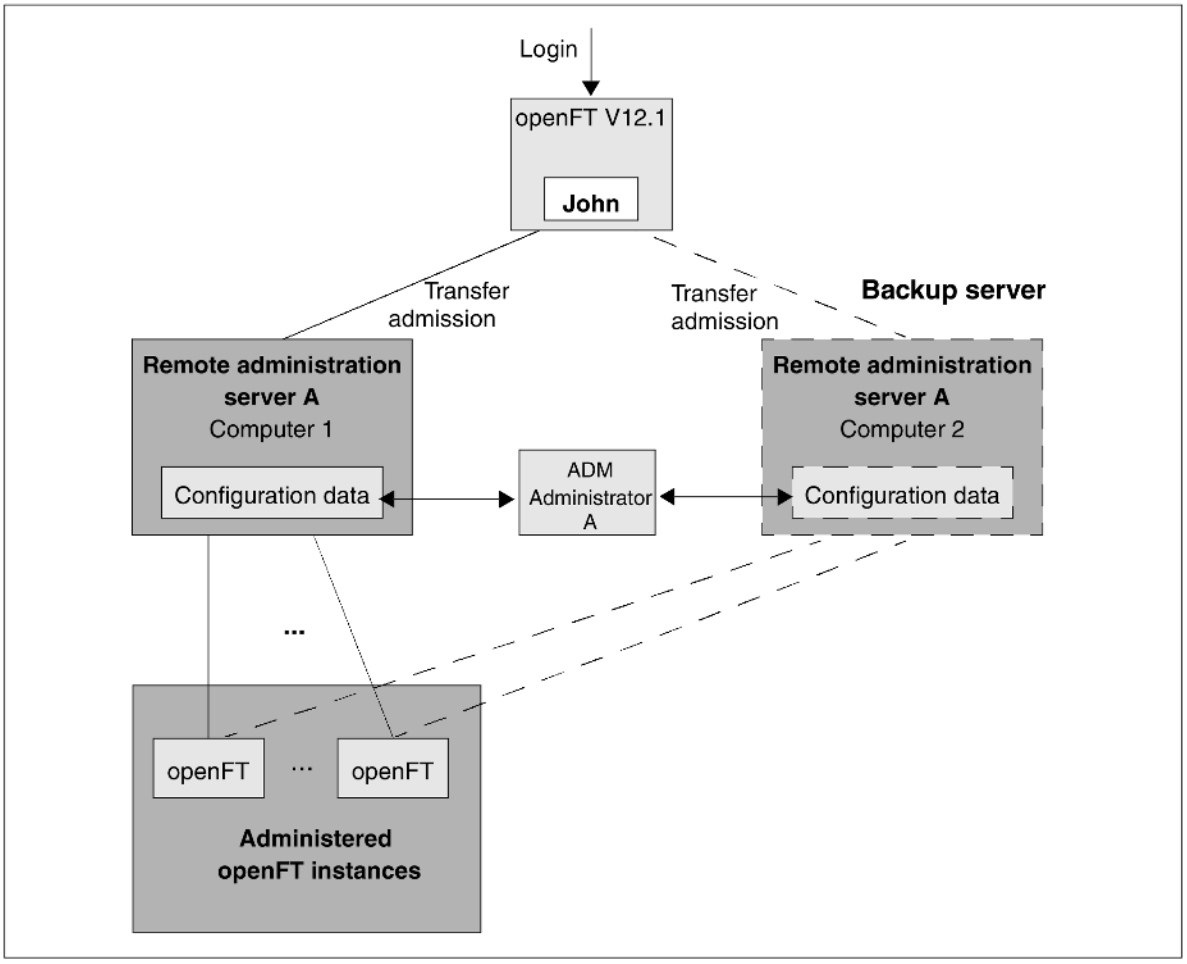
The figure below shows an example of this.



Separate configuration with two remote administration servers

Areas A and B are theoretically strictly separated, but *Jack* is permitted to administer instances from areas A and B, whereas *Curt* can only administer instances from area B.

The same method can also be used to define a redundant configuration with a second remote administration server. This allows implementation of a simple backup solution.



Redundant configuration with a second administration server as a backup.

If Computer 1 fails, the remote administrator can use Computer 2 as the remote administration server. In order to do this,

- the ADM administrator must always ensure that the configuration data on the two computers is consistent,
- the admission profiles for accessing the remote administration server and the partner list entries (if they are used) are identical on Computer 1 and Computer 2,
- the admission profiles on the administered instances are defined in such a way that they accept both remote administration servers as partners.

If authentication is used, you must also note that

- the keys for the computers from which administration is performed must be present on both remote administration servers,
- the administered instances require the keys of both remote administration servers.

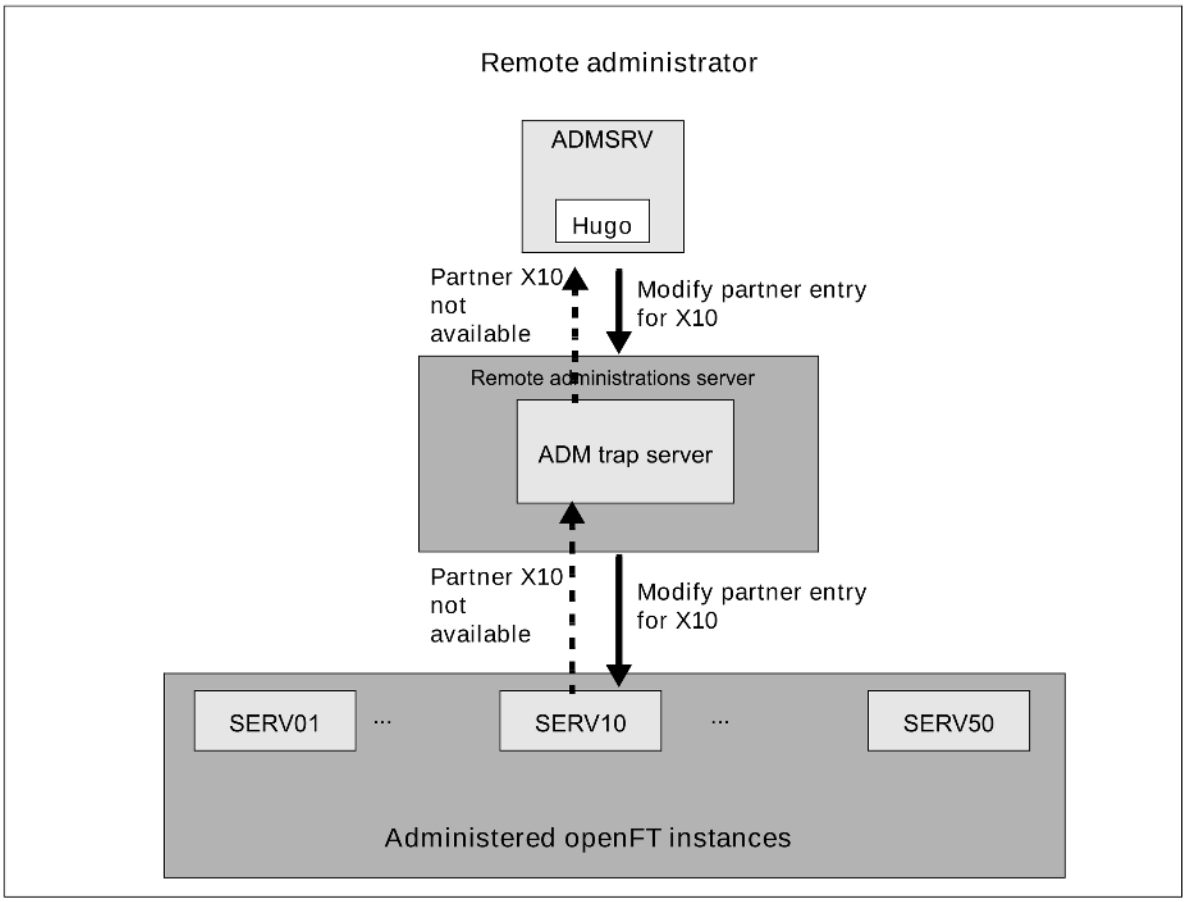
For this reason, with complex configurations in particular, you should implement failsafe protection of the remote administration server using a cluster. You can find examples of how to set up a cluster in the manual "openFT (Unix and Windows systems) - Installation and Operation".

7.4.2 ADM traps

ADM traps are short messages that openFT sends to the **ADM trap server** if certain events occur during operation of openFT. Such events may include errored FT requests, status changes or the unavailability of partners, for instance.

The ADM traps are stored permanently on the ADM trap server. This allows openFT systems to be monitored at a central location. The FT administrator of the ADM trap server is thus provided with a simple way of gaining an overview of events that have occurred on the openFT instances he is monitoring using the openFT Explorer or the *ftshwatp* command.

If the ADM trap server is simultaneously used as a remote administration server, remote administrators can also view traps from other systems and hence monitor the systems that they are administering, see the following example:



ADM trap server on the remote administration server

SERV10 reports via ADM trap that the partner X10 is not accessible. The remote administrator HUGO reads this trap, checks the log records and other items on SERV10 and finds out that the IP address of X10 has changed but that this modification has not been entered in the partner list. Via the remote administration server, HUGO modifies the partner list entry for X10 on SERV10 by means of the corresponding FT command.

8 License provisions

The following provisions apply to the use of *libxml2* and Secure FTP and xerces-J for openFT-Script.

Use of libxml2

libxml2 is used for processing XML data. This contains the XML C Parser and an XML toolkit. *libxml2* was originally developed for the Gnome project, but can also be used outside Gnome. *libxml2* is freeware available under the MIT license:

```
Copyright (c) <2008> <Daniel Veillard>
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies
of the Software, and to permit persons to whom the Software is furnished to do
so, subject to the following conditions:
The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

Use of openSSL for Secure FTP

The following provisions apply to the use of Secure FTP.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/). This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

```
LICENSE ISSUES
==============
The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the
OpenSSL License and the original SSLeay license apply to the toolkit. See below
for the actual license texts. Actually both licenses are BSD-style Open Source
licenses. In case of any license issues related to OpenSSL please contact
openssl-core@openssl.org.
OpenSSL License
---------------
================================================================================
Copyright (c) 1998-2006 The OpenSSL Project. All rights reserved.
Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this
   list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must
   display the following acknowledgment:
   "This product includes software developed by the OpenSSL Project for use in
   the OpenSSL Toolkit. (http://www.openssl.org/)"
4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse
```

```
   or promote products derived from this software without prior written
   permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called "OpenSSL" nor may
   "OpenSSL" appear in their names without prior written permission of the
   OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following
   acknowledgment:
   "This product includes software developed by the OpenSSL Project for use in
   the OpenSSL Toolkit (http://www.openssl.org/)"
THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT
SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
OF THE POSSIBILITY OF SUCH DAMAGE.
================================================================
This product includes cryptographic software written by Eric Young
(eay@cryptsoft.com). This product includes software written by Tim Hudson
(tjh@cryptsoft.com).
Original SSLeay License
-----------------------
Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.
This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).
The implementation was written so as to conform with Netscapes SSL. This library
is free for commercial and non-commercial use as long as the following conditions
 are aheared to. The following conditions apply to all code found in this
distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code.
The SSL documentation included with this distribution is covered by the same
copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).
Copyright remains Eric Young's, and as such any Copyright notices in the code
are not to be removed. If this package is used in a product, Eric Young should
be given attribution as the author of the parts of the library used. This can
be in the form of a textual message at program startup or in documentation
(online or textual) provided with the package.
Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the copyright notice, this list
   of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must
   display the following acknowledgement:
   "This product includes cryptographic software written by Eric Young
   (eay@cryptsoft.com)"
   The word 'cryptographic' can be left out if the rouines from the library
   being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the
   apps directory (application code) you must include an acknowledgement:
   "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANT-
ABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL
THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
```

```
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSI-
BILITY OF SUCH DAMAGE.
```

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i. e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

Use of xerces-J for openFT-Script

The following provisions apply to operation with openFT-Script.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

```
/* ====================================================================
 * The Apache Software License, Version 1.1
 *
 * Copyright (c) 2000 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Apache" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
```

```
* THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
* OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
* DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
* USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
* ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
* OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
* OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
* ====================================================================
*
* This software consists of voluntary contributions made by many
* individuals on behalf of the Apache Software Foundation.  For more
* information on the Apache Software Foundation, please see
* <http://www.apache.org/>.
*
* Portions of this software are based upon public domain software
* originally written at the National Center for Supercomputing Applications,
* University of Illinois, Urbana-Champaign.
*/
```

9 Glossary

Italic type indicates a reference to other terms in this glossary.

ABEND

Abnormal termination of program in z/OS.

absolute path name

The entire path name, from the root directory to the file itself.

access control

FTAM specific file attribute in the *virtual filestore*, attribute of the *security group* that defines *access rights*.

access protection

Comprises all the methods used to protect a data processing system against unauthorized system access.

access right / access admission

Derived from the *transfer admission*. The access right defines the scope of access for the user who specifies the transfer admission.

ACF-2

Program product from Computer Associates for system and data access control on z/OS.

action list

Component of the FTAM specific file attribute *access control* (attribute of the *security group*) in the *virtual filestore* that defines *access rights*.

ADM administrator

Administrator of the *remote administration server*. This is the only person permitted to modify the configuration data of the remote administration server.

ADM partner

Partner system of an openFT instance with which communication takes place over the *FTADM protocol* in order to perform *remote administration*.

ADM traps

Short messages sent to the *ADM trap server* if certain events occur during operation of openFT.

ADM trap server

Server that receives and permanently stores the *ADM traps*. It must be configured as a *remote administration server*.

administrated openFT instance

openFT instances that are able to be administered by *remote administrators* during live operation.

admission profile

Way of defining the *FTAC* protection functions. Admission profiles define a *transfer admission* that has to be specified in *FT requests* instead of the *LOGON* or *Login authorization*. The admission profile defines the *access rights* for a user ID by restricting the use of parameters in *FT requests*.

admission profile, privileged

see *privileged admission profile*

admission set

In *FTAC*, the admission set for a particular user ID defines which FT functions the user ID may use and for which *partner systems*.

admission set, privileged

see *privileged admission set*

AES (Advanced Encryption Standard)

The current symmetrical encryption standard, established by NIST (National Institute of Standards and Technology), based on the Rijndael algorithm, developed at the University of Leuven (B). The openFT product family uses the AES method to encrypt the request description data and possibly also the file contents or the file and directory list attributes.

alphanumeric

Alphanumeric characters comprise alphabetic and numeric characters, i.e. the letters A-Z and the digits 0-9 and, on z/OS, the additional characters $, @, #.

AMODE

Specification for addressing a module (24-bit or 31-bit addresses) in z/OS.

ANSI code

Standardized 8-bit character code for message exchange. The acronym stands for "American National Standards Institute".

API (Application Programming Interface)

An interface that is freely available to application programmers. It provides a set of interface mechanisms designed to support specific functionalities.

Application Entity Title (AET)

The Application Entity Title consists of Layer 7 addressing information of the *OSI Reference Model*. It is only significant for *FTAM partners*.

asynchronous request

Once the *FT request* has been submitted, it is processed independently of the user. The user can continue working once the system has confirmed acceptance of the request. (see also *synchronous request*).

audit

Fundamental function of a secure system; logging of operating sequences and preparation of the logged data.

authentication

Process used by openFT to check the unique identity of the request partner.

background process

A process in Unix systems that runs independently of the user process. A background process is started by placing the special character & at the end of a command. The process which initiates the background process is then immediately free for further tasks and is no longer concerned with the background process, which runs simultaneously.

basic functions

Most important file transfer functions. Several basic functions are defined in the *admission set* which can be used by a login name. The six basic functions are:

- inbound receive
- inbound send
- inbound follow-up processing
- inbound file management
- outbound receive
- outbound send

central administration

Central administration in openFT incorporates the *remote administration* and *ADM traps* functions and requires the use of a *remote administration server*.

character mode

File names, pre, post and follow-up processing as well as remote commands are seen in their character presentation. An Ä in a remote file name is for example understood in character mode in the partner system also as an Ä, even if different system coding is set there and different local coding is set.

character repertoire

Character set of a file in the *virtual filestore.*
In the case of files transferred with *FTAM partners* it is possible to choose between: *GeneralString* **,** *GraphicString* **,** *IA5String* **and** *VisibleString* **.**

Character Separated Values

see *Comma Separated Values.*

client

- Term derived from client/server architectures: the partner that makes use of the services provided by a *server*.
- Logical instance which submits requests to a *server.*

cluster

A number of computers connected over a fast network and which in many cases can be seen as a single computer externally. The objective of clustering is generally to increase the computing capacity or availability in comparison with a single computer.

Comma Separated Values (CSV)

This is a quasi-tabular output format that is very widely used in the PC environment in which the individual fields are separated by a separator (often a semicolon ";"). It permits the further processing of the output from the most important openFT commands using separate tools.

communication computer

Computer for constructing a *data communication system*.

communication controller

see *preprocessor*

compression

This means that several identical successive characters can be reduced to one character and the number of characters is added to this. This reduces transfer times.

computer network, open

see *open computer network*

concurrency control

Component of the FTAM file attribute *access control* (part of the *security group*) in the *virtual filestore* that controls concurrent access. openFT (BS2000) offers only passive and partial support for concurrency control. Note: "partial support" is a technical term taken from the FTAM environment that means that the parameter is interpreted correctly at the syntactic level but is not genuinely supported.

configuration user ID

Each openFT instance in a BS2000 system requires an ID, on which the variable files of this file are stored (for the standard instance: $SYSFJAM).

connectivity

In general, the ability of systems and partners to communicate with one another. Sometimes refers simply to the communication possibilities between transport systems.

constraint set

Component of the FTAM file attribute *document type*.

contents type

FTAM specific file attribute in the *virtual filestore,* attribute of the *kernel group* that describes the file structure and the form of the file contents.

cross domain connection

A connection between computers that are located in different SNA domains.
A cross domain connection from a TRANSDATA network (as a SNA domain) to an SNA network requires the software product TRANSIT-CD to be used as a gateway.

cross network connection

A connection between computer that are located in different SNA networks.
A cross network connection from a TRANSDATA network to one or more SNA networks requires the software product TRANSIT-CD and, depending on the configuration, may also require TRANSIT-SNI to be used as a *gateway*.

DASD (Direct Access Storage Device)

Disk storage in BS2000 systems.

data communication system

Sum of the hardware and software mechanisms which allow two or more communication partners to exchange data while adhering to specific rules.

data compression

Reducing the amount of data by means of compressed representation.

data encoding

Way in which an *FT system* represents characters internally.

Data Encryption Standard (DES)

International data encryption standard for improved security. The DES procedure is used in the FT products to encrypt the request description data and possibly the request data or the file and directory list attributes if connections are established to older versions of openFT that do not support *AES*.

data protection

- In the narrow sense as laid down by law, the task of protecting personal data against misuse during processing in order to prevent the disclosure or misappropriation of personal information.
- In the wider sense, the task of protecting data throughout the various stages of processing in order to prevent the disclosure or misappropriation of information relating to oneself or third parties.

data security

Technical and organizational task responsible for guaranteeing the security of data stores and data processing sequences, intended in particular to ensure that

- only authorized personnel can access the data,
- no undesired or unauthorized processing of the data is performed,
- the data is not tampered with during processing,
- the data is reproducible.

data set

File in z/OS.

DHCP

Service in TCP/IP networks that automatically assigns IP addresses and TCP/IP parameters to clients on request.

Direct Access Storage Device (DASD)

Disk storage device in BS2000 systems.

directory

Directories are folders in the hierarchical file system of a Unix system (including POSIX) or a Windows system that can contain files and/or further directories.
In BS2000 (DVS), PLAM libraries are interpreted as directories.
openFT (z/OS) interprets, on the one hand, the contents of a PO or PDSE data set (and the members included in it) as a directory, and on the other hand also all files with a common name up to a qualifying delimiter (dot).

document type

Value of the FTAM specific file attribute *contents type* (attribute of the *kernel group*). Describes the type of file contents in the *virtual filestore*.

- *document type* for text files: FTAM-1
- *document type* for binary files: FTAM-3

dynamic partner

partner system that is either not entered in the *partner list* (*free dynamic partner*) or that is entered in the partner list with only address but without a name (*registered dynamic partner*).

EBCDIC

Standardized code for message exchange as used in BS2000 or z/OS systems.
The acronym stands for "Extended Binary Coded Decimal Interchange Code".

emulation

Components that mimic the properties of another device.

encoding mode

Mode for encoding file names, pre, post and follow-up processing as well as remote commands, see also *transparent mode* and *character mode*.

entity

see *instance*

Explorer

A program from Microsoft that is supplied with Windows operating systems to facilitate navigation within the file system.

file attributes

A file's properties, for example the size of the file, access rights to the file or the file's record structure.

file directory / file catalog

File in a BS2000 system present in every *pubset* (in SM pubsets there is a file directory in every volume set). All a pubset's files and job variables are entered in the corresponding *file directory*. Files on private disks and tapes can be entered in the file directory.
A catalog entry contains all a file's or job variable's attributes (protection attributes, location of the administered data etc.

file management

Possibility of managing files in the remote system. The following actions are possible:

- Create directories
- Display and modify directories
- Delete directories
- Display and modify file attributes
- Rename files
- Delete files.

file processing

The openFT "file processing" function makes it possible to send a receive request in which the output of a remote command or program is transferred instead of a remote file.

filestore, virtual

see *virtual filestore*

file transfer request

see *FT- request*

firewall processor

Processor which connects two networks. The possible access can be controlled precisely and also logged.

fixed-length record

A record in a file all of whose records possess the same, agreed length. It is not necessary to indicate this length within the file.

follow-up processing

FT function that initiates execution of user-specified commands or statements in the *local* and/or the *remote system* after an *FT request* has been completed. The user may define different follow-up processing, depending on the success or failure of FT request processing. See also *preprocessing* and *postprocessing*.

follow-up processing request

Statements contained within an *FT request* which perform *follow-up processing* after file transfer.

free dynamic partner

Partner system that is not entered in the partner list.

FT administrator

Person who administers the openFT product installed on a computer, i.e. who is responsible, among other things, for the entries in the *partner list* as well as for controlling resources. On Unix systems, openFT can be administered from all login names with UID=0.

FT profile

See *admission profile*.

FT request

Request to an *FT system* to transfer a file from a *sending system* to a *receive system* and (optionally) start *follow-up processing requests*.

FT system

System for transferring files that consists of a computer and the Software required for file transfer.

FT trace

Diagnostic function that logs FT operation.

FTAC (File Transfer Access Control)

Extended access control for file transfer and file management. In the case of BS2000 and z/OS, this is implemented by means of the product openFT-AC, for other operating systems it is a component of the openFT product, e.g. in openFT (Unix systems) or openFT (Windows).

FTAC administrator

Administrator of the FTAC functions; should be identical to the person responsible for data security in the system.
The FTAC administrator specifies for their system, among other things, the security-technical framework in the form of a standard admission set that is valid for all users.

FTAC logging function

Function which FTAC uses to log each access to the protected system via file transfer.

FTAC transfer admission

Authorization for file transfer and file management when using FTAC. The transfer admission is then used in place of the *LOGON or LOGIN authorization*.

FTADM protocol

Protocol used for communication between two openFT instances in order to perform *remote administration* or transfer *ADM traps*.

FTAM-1

document type for text files

FTAM-3

document type for binary files

FTAM catalog

The FTAM catalog is used on Unix and Windows systems to extend the file attributes available in Unix systems. It is only relevant for access using FTAM. For example, a file can be deleted using the command *rm* on a Unix system or *erase* on a Windows system, even if the *permitted actions* parameter does not allow this.

FTAM file attributes

All systems which permit file transfer via FTAM protocols must make their files available to their partners using a standardized description (ISO 8571). To this end, the attributes of a file are mapped from the physical filestore to a *virtual filestore* and vice versa. This process distinguishes between three groups of file attributes:

- kernel group: describes the most important file attributes.
- storage group: contains the file's storage attributes.
- security group: defines security attributes for file and system access control.

FTAM partner

Partner system that uses *FTAM protocols* for communication.

FTAM protocol (File Transfer, Access and Management)

Protocol for file transfer standardized by the "International Organization for Standardization" (ISO) (ISO 8571, FTAM).

FTP partner

Partner system that uses *FTAM protocols* for communication.

FTP protocol

Manufacturer-independent protocol for file transfer in TCP/IP networks.

functional standard

Recommendation defining the conditions and the forms of application for specific ISO standards (equivalent term: *profile*). The transfer of unstructured files is defined in the European Prestandard CEN/CENELEC ENV 41 204; file management is defined in the European Prestandard CEN /CENELEC ENV 41205.

gateway

Generally understood to mean a computer that connects two or more networks and which does not function as a bridge. Variants: gateway at network level
(i.e. router or OSI relay), transport and application gateway.

gateway processor

Communication computer that links a computer network to another computer network. The mapping of the different protocols of the various computer networks takes place in gateway processors.

Generalized Trace Facility (GTF)

IBM tool for generating traces (in particular for monitoring the data traffic between an application program and the relevant VTAM applications and between VTAM applications and the data communication line).

general string

Character repertoire for files transferred to and from *FTAM partners*.

global privileges

All the privileges that can be assigned using the BS2000 command SET-PRIVILEGE including the security administrator privilege and the TSOS privilege. *Global privileges and system administrator privileges* are identical.

global request identification / **global request ID** / **global request number**

Request number that the *initiator* of an openFT or FTAM request transfers to the *responder*. This means that the global request ID in the responder is identical to the *request ID* in the initiator. The responder generates its own (local) request ID for the request. This means that information stored in both the initiator and the responder can be unambiguously assigned to a request. This is particularly important if the request has to be restarted.

global user administration

In BS2000 systems, this comprises the administration of user IDs and user groups and covers resources and user rights, the creation, modification and deletion of user IDs and user groups

GraphicString

Character repertoire for files transferred to and from *FTAM partners*.

guard

A component of the GUARDS condition administration system in BS2000 systems. A guard unites conditions which are evaluated by the standard GUARDS condition administration system on request.

GUARDS (Generally Usable Access Control Administration System)

Object administration for *Guards*.

heterogeneous network

A network consisting of multiple subnetworks functioning on the basis of different technical principles.

homogeneous network

A network constructed on the basis of a single technical principle.

host

Formerly a large-scale data processing system which required a *front-end processor* in order to be able to communicate. Nowadays, the term used for BS2000 or z/OS systems.

HOSTS file

Network administration file in Unix and Windows systems that contains the Internet addresses, the processor names and the alias names of all accessible computers.

IA5String

Character repertoire for files transferred to and from *FTAM partners.*

identification

Procedure making it possible to identify a person or object.

IEBCOPY

IBM tool for copying libraries (PO or PDSE data sets).

IEBGENER

IBM tool for copying sequential files (PS data sets).

IEBPTPCH

IBM tool for printing files.

inbound file management

Request issued in a *remote system* for which directories or file attributes of the *local system* can be displayed, file attribute modified or local file deleted.

inbound follow-up processing

Request issued in a *remote system* with *follow-up processing* in the *local system*.

inbound receive

Request issued in the *remote system*, for which a file is received in the *local system*.

inbound request / **inbound submission**

Request issued in another system.

inbound send

Request issued in a *remote system* for which a file is sent from the *local system* to the remote system.

initiator

Here: *FT system* that submits an *FT request*.

instance / **entity**

A concept of OSI architecture: active element in a layer. Also see *openFT instance.*

instance ID

A network-wide, unique address of an openFT instance.

integrity

Unfalsified, correct data following the processing, transfer and storage phases.

Interactive Problem Control System (IPCS)

IBM tool for formatting a machine-readable (unformatted) dump.

interoperability

Capability of two *FT systems* to work together.

ISO/OSI reference model

The ISO/OSI Reference Model is a framework for the standardization of communications between open systems. (ISO=International Standards Organization).

ISPF, ISPF/PDF

Menu-driven utilities in z/OS for software development and for conducting a (TSO) dialog.

job

Sequence of commands, statements and data.
On z/OS: A sequence of JCL statements (batch).

job class

Job classes combine *jobs* in BS2000 systems which share certain properties and characteristics.

job transfer

Transfer of a file that constitutes a *job* in the *receive system* and is initiated as a job there.

joinfile / **user catalog** / **user ID catalog**

File in a BS2000 system that contains the *user attributes* of all the *user IDs* in a *pubset*.

kernel group

Group of FTAM specific file attributes of the *virtual filestore* that encompasses the kernel attributes of a file.

library

File with internal structure (members)

library member

Part of a library. A library member may in turn be subdivided into a number of records.

Local Area Network (LAN)

Originally a high-speed network with limited physical extension. Nowadays, any network, that uses CSMA/CD, Token Ring or FDDI irrespective of the range **(see also** *WAN Wide Area Network*).

local system

The *FT system* at which the user is working.

logging function

Function used by openFT to log all file transfer accesses to the protected system.

logging record

see *log record*.

log record

Contains information about access checks performed by openFT (FTAC log record) or about a file transfer or remote administration request which is started when the access check was successful (FT log record or ADM log record).

Logical Unit (LU)

Interface between an application program and the SNA data communications network. The LU type describes the communications characteristics.

Login authorization

Access authorization to a computer. The login authorization which (as a rule) consists of the login name and the password authorizes dialog operation, see also *LOGON authorization*.

LOGON authorization

Access authorization to a computer. The LOGON authorization which (as a rule) consists of user ID, account number and password authorizes the user to make use of interactive operation.

mailbox

The mailbox is a file in a Unix system which is read using the mail command.
Each user has a mailbox for receiving messages.

mainframe

Computer (consisting of one or more processors) which runs under the control of a universal operating system (e.g. BS2000 or z/OS).
Synonyms: BS2000 computer, host computer.

maximum-string-length

Specifies the maximum length of *strings* within a file in the *virtual FTAM filestore*.

named partner

partner system entered by its name in the *partner list*.

Network Control Program (NCP)

Operating system of the front-end-processor for SNA hosts.

NEA

Name of a network architecture in BS2000 systems.

NetMaster

Tool for controlling a data communication system.

NetView

IBM tool for controlling a data communication system.

network description file

File used up to openFT V9 that contains specifications concerning *remote systems* (*FT systems*).

Network Management Kernel

Component of the Network Management Platform in BS2000 systems; responsible for forwarding network management requests as well as for centralized tasks such as logging, authorization checks, request and application administration.

offline logging

The log file can be changed during operation. Following this changeover, the previous log file is retained as an offline log file; new log records are written to a new log file. It is still possible to view the log records in an offline log file using the tools provided by openFT.

open computer network

Computer network in which communication is governed by the rules of ISO/OSI. Interoperation of different computers from various vendors is made possible by defined *protocols*.

openFT Explorer

openFT program for Unix and Windows systems that provides a graphical user interface that allows file transfer and administration functions to be performed.

openFT installation directory

Path under which openFT (Windows) is installed. This path can be freely selected during interactive installation. It can be set with the INSTALLDIR parameter during unattended installation. The default path depends on the language setting and the version of the Windows operating system. (Default: % Program Files%\openFT).

openFT instance

Several openFT systems, so-called openFT instances, can be running simultaneously on an individual computer or a cluster of a TCP/IP network on the HIPLEX cluster on a Sysplex cluster. Each instance has its own address (instance ID, virtual BCAM host in BS2000 systems, host in z/OS systems) and is comprised of the loaded code of the openFT products (including add-on products if they are available) and of the variable files such as the partner list, logging files, key library, request queue, etc.

openFT Monitor

Program for Unix and Windows systems that allows the monitoring data for openFT operation to be shown in the form of a chart. openFT Monitor requires a graphics-capable terminal.

openFT partner

Partner system which is communicated with using *openFT protocols*.

openFT protocols

Standardized *protocols* for file transfer (SN77309, SN77312).

openFT-FTAM

Add-on product for openFT (for BS2000, Unix systems and Windows systems) that supports file transfer using FTAM protocols. FTAM stands for File Transfer, Access and Management (ISO 8571).

openFT-Script

openFT interface for Unix and Windows systems providing an XML based script language that includes file transfer and file management functions. This interface allows you to combine several file transfer or file management requests to form a single openFT-Script request.

openFT-Script commands

Commands used for administering openFT-Script requests in Unix and Windows systems.

operating parameters

Parameters that control the *resources* (e.g. the permissible number of connections).

outbound request / outbound submission

Request issued in your own processor.

outbound receive

Request issued locally for which a file is received in the *local system*.

outbound send

Request issued locally for which a file is sent from the *local system*.

owner of an FT request

Login name in the *local system* or *remote system* under which this *FT request* is executed. The owner is always the ID under which the request is submitted, not the ID under which it is executed.

partitioned data set extended (PDSE data set)

Library in the IBM z/OS Data Management System. Contains individual members and can be used instead of a partitioned organized data set. The IBM software product "Data Facility Storage Management Subsystem" (DFSMS) is required to use PDSE.

partitioned organized data set (PO data set)

Library of the IBM z/OS Data Management System. Contains individual members.

partner

see *partner system*

partner list

File containing specifications concerning *remote systems* (*FT systems*).

partner system

Here: *FT system* that carries out *FT requests* in cooperation with the *local system*.

password

Sequence of characters that a user must enter in order to access a user ID, file, job variable, network node or application. The user ID password serves for user *authentication*. It is used for access control. The file password is used to check access rights when users access a file (or job variable). It is used for file protection purposes.

permitted actions

FTAM specific file attribute in the *virtual filestore*; attribute of the *kernel group* that defines actions that are permitted in principle.

Personal Audit for Individual Accountability

Trace of individual system utilization. Identification can take the following forms:

- a user ID corresponds to a user, or
- a user may use only one operator terminal.

physical sequential data set / **PS data set**

Sequential file in the IBM z/OS Data Management System; similar to a BS2000 SAM file.

Physical Unit (PU)

Each node of an SNA network contains a Physical Unit (PU) as an addressable instance. This is responsible for monitoring the connection to the host and for monitoring the *Logical Units* (LUs).

port number

Number that uniquely identifies a TCP/IP application or the end point of a TCP/IP connection within a processor.

POSIX (Portable Open System Interface)

Board and standards laid down by it for interfaces that can be ported to different system platforms.

postprocessing

openFT makes it possible to process the received data in the receiving system through a series of operating system commands. Postprocessing runs under the process control of openFT (in contrast to *follow-up processing*).

preprocessing

The preprocessing facility in openFT can be used to send a receive request in which the outputs of a remote command or program are transferred instead of a file. This makes it possible to query a database on a remote system, for example. Preprocessing also may be issued locally.

preprocessor / **communication controller**

A processor system connected upstream of the mainframe (BS2000 or z/OS system) which performs special communication tasks in the network. Synonym: communication processor.

presentation

Entity that implements the presentation layer (layer 6) of the *ISO/OSI Reference Model* in an *FT system* that uses e.g. *FTAM protocols*.

presentation selector

Subaddress used to address a *presentation application*.

private key

Secret decryption key used by the recipient to decrypt a message that was encrypted using a *public key*. Used by a variety of encryption procedures including the *RSA procedure*.

privilege

- In BS2000 and z/OS systems: Global privilege within the system that authorizes a user to execute certain commands and call certain program interfaces (e.g. TSOS privilege).
- In BS2000 systems: Set of user-specific attributes that are used by the access control system.

privileged admission profile

Admission profile that allows the user to exceed the *FTAC administrator's* presettings in the *admission set*. This must be approved by the *FTAC administrator* who is the only person able to privilege admission profiles.

privileged admission set

Admission set belonging to the *FTAC administrator*.

procedure

In z/OS: command procedure, corresponds in principle to an IBM CLIST or REXX procedure.

profile

In OSI, a profile is a standard which defines which protocols may be used for any given purpose and specifies the required values of parameters and options.
Here: a set of commands assigned to a user ID. The permissibility of These commands is ensured by means of syntax files. See also *admission profile*, *privileged admission profile*.

prompting in procedures

Function in Unix and Windows systems used to prompt the user at the terminal to enter data required to run the procedure.

protocol

Set of rules governing information exchange between peer partners in order to achieve a defined objective. This usually consists of a definition of the messages that are to be exchanged and the correct sequencing of Messages including the handling of errors and other exceptions.

public key

Public encryption key defined by the receiver of a message, and made public or made known to the sender of the message. This allows the sender to encrypt messages to be sent to the receiver. Public keys are used by various encryption methods, including the *Rivest Shamir Adleman (RSA) procedure*. The public key must match the *private key* known only to the receiver.

public space

Named disk storage area in BS2000 systems which is available to a defined number of user IDs within the operating system. This storage area may be located on one or more Public Volume Sets (*pubsets*).

pubset / public volume set

Set of shared, named disk storage units in a BS2000 system which is defined by a catalog identification (catid). A distinction is made between *SF pubsets* and SM pubsets.

RACF

IBM product for system and data access control.

RAS

Remote Access Service; a Windows service that enables communication with remote systems.

receive file

File in the *receive system* in which the data from the *send file* is stored.

receive system

System to which a file is sent. This may be the *local system* or the *remote system*.

record

Set of data that is treated as a single logical unit.

registered dynamic partner

Partner system that is entered in the partner list with only an address but no name.

relative path name

The path from the current *directory* to the file.

relay

OSI term for an element in a layer that acts as an intermediary between two other partners and thus makes communications between these two partners possible. In the narrow sense, on the network layer a relay is the functional equivalent of a *router*.

relay program

Program in a *gateway processor* that maps the different protocols onto one another.

remote administration

Administration of openFT instances from remote computers.

remote administration server

Central component required for *remote administration* and for *ADM traps*. A remote administration server runs on a Unix or Windows system running openFT as of V11.0. If it is used for *remote administration*, it contains all the configuration data required for this purpose.

remote administrator

Role configured on the *remote administration server* and which grants permission to execute certain administration functions on certain openFT instances.

remote system

see *partner system*

request

see *FT request,*

request file

see *request queue*.

request queue

File containing *asynchronous requests* and their processing statuses.

request identification / **request ID**

Number assigned by the local system that identifies an *FT request.*

request management

FT function responsible for managing *FT requests*; it ensures request processing from the submission of a request until its complete processing or termination.

request number

see *request identification*

request storage

FT function responsible for storing *FT requests* until they have been fully processed or terminated.

resources

Hardware and software components needed by the *FT system* to execute an *FT request* (*tasks* processes, connections, lines). These resources are controlled by the *operating parameters*.

responder

Here: *FT system* addressed by the *initiator*.

restart

Automatic continuation of an *FT request* following an interruption.

restart point

Point up to which the data of the *send file* has been stored in the *receive file* when a file transfer is interrupted and at which the transfer of data is resumed following a *restart*.

result list[ing]

In BS2000 and z/OS systems: List with information on a completed file transfer.
This is supplied to the user in the *local system* and contains information on his or her *FT requests*.

REXX

IBM procedure language.

RFC (Request for Comments)

Procedure used on the Internet for commenting on proposed standards, definitions or reports. Also used to designate a document approved in this way.

RFC1006

Supplementary protocol for the implementation of ISO transport services (transport class 0) using TCP/IP.

Rivest-Shamir-Adleman-procedure (RSA procedure)

Encryption procedure named after its inventors that operates with a key pair consisting of a *public key* and a *private key*. Used by the openFT product family in order to reliably check the identity of the partner system and to transmit the AES key to the partner system for encrypting the file contents or the file and directory list attributes.

router

Network element that is located between networks and guides message flows through the networks while simultaneously performing route selection, addressing and other functions. Operates on layer 3 of the OSI model.

RPC (Remote Procedure Call)

Cross-network server procedure call issued by client.

security attributes

An object's security attributes specify how and in what ways the object may be accessed.

Secure FTP

Method by which a connection is tunneled using the *FTP protocol*, thus allowing secure connections with encryption and *authentication*.

security group

Group of FTAM specific file attributes in the *virtual filestore*, encompassing the security attributes of a file.

security level

When FTAC or *FTAC functions* are used, the security level indicates the required level of protection against a *partner system*.

send file

File in the *sending system* from which data is transferred to the *receive file*.

sending system

Here: *FT system* that sends a file. This may be the *local system* or the *remote system*.

server

Logical entity or application component which executes a client's requests and assures the (coordinated) usage of all the generally available services (File, Print, data base, Communication, etc.). May itself be the client of another server.

service

- As used in the OSI architecture: a service is the set of functions that a service provider makes available at a service access point.

- As used in the client/server architecture: a set of functions that a Server makes available to its clients.
- Term used in Unix and Windows systems: A program, routine or process used to perform a particular system function to support other programs, in particular on a low level (hardware-related).

service class

Parameter used by *FTAM partners* to negotiate the functions to be used.

session

- In OSI, the term used for a layer 5 connection.
- In SNA, a general term for a connection between communication partners (applications, devices or users).

session selector

Subaddress used to address a *session* application.

SF pubset (Single Feature Pubset)

One or more disks in a BS2000 system whose key properties (disk format, allocation unit) match and which are used to store files and JVs under a shared catalog ID.

shell metacharacters

On Unix and Windows systems, the following metacharacters have special meanings for the Unix shell or Windows command prompt: *, [ ], ?, <, >, |, &, &&, ( ), { }

SMF (Service Management Facility)

Tool for controlling services on Solaris systems.

SMF (System Management Facility)

IBM Tool for collecting accounting data and statistics.

SMP/E (System Modification Program/Extended)

IBM product used to install and manage the software products, their versions and corrections.

SNA network

Data communication system that implements the Systems Network Architecture (SNA) of IBM.

SNMP (Simple Network Management Protocol)

Protocol for TCP/IP networks defined by the Internet Engineering Task Force (IETF) for the transfer of management information.

special characters

see *shell metacharacters.*

Standard Access Control

BS2000 systems: Consists of the ACCESS and USER-ACCESS rights that are defined in the CREATE-FILE or MODIFY-FILE-ATTRIBUTES commands.

standard admission set

This standard admission set applies by default to all users for whom there is no dedicated admission set. These default settings may be restricted further by the user for his or her own admission set.

standard error output (stderr)

By default, standard error output on Unix and Windows systems is to the screen.

standard input (stdin)

By default, standard input on Unix and Windows systems is from the keyboard.

standard instance

The first openFT-instance that is always available and is activated or loaded when starting openFT. By default all openFT commands refer to this instance, if no other instance was specified.

standard output (stdout)

By default, standard output on Unix and Windows systems is to the screen.

storage group

FTAM specific file attribute in the *virtual filestore*, encompasses the storage attributes of a file.

string

Character string

string significance

Describes the format of *strings* in files to be transferred using *FTAM protocols*.

subsystem

In BS2000 systems: Part of a system which processes a self-contained group of functions.

SU Privilege

Privilege of an FTAC administrator in z/OS. This privilege allows the administrator to set up admission profiles for which TRANSFER-ADMISSIONS have been released on other user IDs without the need to know the current password. This privilege is defined in the FTACADM member of the parameter library.

synchronous request

The user task (user process) that submitted the *FT request* waits for transfer to terminate. The user cannot continue working (see also *asynchronous request*).

SYSFILE environment

System files in BS2000 systems; the SYSFILE environment designates the totality of the system files assigned to a request.

system

see *FT- system*

system, local

see *local system*

system, remote

see *remote system*

system administration

- Structural unit in the computer center
- Group of individuals who employ user IDs that are associated with global privileges.

system administrator command

Command which cannot be submitted by any user ID but only by user IDs which possess the corresponding (global) privileges (in BS2000 systems by the TSOS user ID, for example).

system administrator privileges

see *global privileges*

system files

The system input/output files assigned to a request in a BS2000 system. Users can only access system files indirectly by means of the SYSFILE command. System files provide data and resources that are required for the functions of the control program.
System files and their primary allocations:

- SYSOUT: output of system messages to terminals
- SYSLST: output of compilation logs etc.via printer(automatic SPOOLOUT)
- SYSLSTnn: as SYSLST; 1 <= nn <= 99; each of the max.99 system filesmust be assigned to a cataloged file
- SYSOPT: output file as SYSLST
- SYSCMD: used to submit commands to the control program
- SYSDTA: used to enter data or statements

system resources

Resources in a computer system that can be requested or released by a *job* or a *task*/*process*.

task

Entity responsible for processes. In BS2000 tasks are used, among other things, to process user jobs (e.g. batch jobs, interactive jobs), see *job*.
In z/OS: Entity responsible for executing one or more programs within a *job*.

TCP/IP (Transmission Control Protocol / Internet Protocol)

Widely used data transmission protocol (corresponds approximately to layers 3 and 4 of the *ISO/OSI reference model*, i.e. network and transport layers); originally developed for the ARPANET (computer network of the US Ministry of Defense) it has now become a de-facto standard.

Top Secret

Program authored by the company Computer Associates for data and system access control.

transfer admission

Short designation for *FTAC transfer admission*.

transfer unit

In an FTAM environment, the smallest data unit for transporting file contents. For *FTAM-1* and *FTAM-3* these are *strings*. A transfer unit can, but need not, correspond to one file record.

Transmission Control Protocol / Internet Protocol

see *TCP/IP*

TranSON

TranSON is a software product that permits secure access to a server. The use of TranSON is transparent to the application. The connection to the remote partner goes from the workstation through a client proxy and server proxy to the remote partner. The client proxy is located on the workstation, and the server proxy is located on the remote partner. The data transferred between the client proxy and the server proxy is encrypted.

transparent mode

File names, pre, post and follow-up processing as well as remote commands are seen in a fixed binary code, independent of local character code settings. Code transformation merely takes place between EBCDIC DF.04-1 (BS2000), IBM1047 (z/OS) and ISO8859-1 (Unix, Windows).

transport connection

Logical connection between two users of the transport system (terminals or applications).

transport layer

Layer 4 of the *ISO/OSI reference model* on which the data transport protocols are handled.

Transport Name Service (TNS)

Service on Unix and Windows systems used to administer properties specific to transport systems. Entries for *partner systems* receive the information on the particular *transport system* employed.

transport protocol

Protocol used on the *transport layer*

transport selector (T-selector)

Subaddress used to address an ISO-8072 application in the *transport layer*.

transport system

- The part of a system or architecture that performs approximately the functions of the four lower OSI layers, i.e. the transport of messages between the two partners in a communication connection.
- Sum of the hardware and software mechanisms that allow data to be transported in computer networks.

TSN (Task Sequence Number)

Identification of a BS2000 process (*task*).

Unicode

The universal character encoding, maintained by the Unicode Consortium. This encoding standard provides the basis for processing, storage and interchange
of text data in any language in all modern software and information technology protocols. The Unicode Standard defines three Unicode encoding forms:
UTF-8, UTF-16 and UTF-32.

universal-class-number

Parameter of the *document-type* that defines the *character-repertoire* of a file to be transferred.

UNIX ®

Registered trademark of the Open Group for a widespread multiuser operating system. A system may only bear the name UNIX if it has been certified by the Open Group.

Unix system

Commonly used designation for an operating system that implements functions typical of UNIX® and provides corresponding interfaces. POSIX and Linux are also regarded as Unix systems.

user

Represented by a *user ID*. The term “user” is a synonym for individuals, applications, procedures etc. which can obtain access to the operating system via a user ID.

user administration

see *global user administration*

user attributes

All the characteristics of the *user ID* in a BS2000 system that are stored in the *joinfile*.

user catalog / **user ID catalog**

see *joinfile*.

user command

Command in a BS2000 system that can be issued under any *user identification* in system mode (/) or in program mode by means of a CMD macro.

user identification / **user ID**

In BS2000 systems: A name with a maximum length of eight characters which is entered in the joinfile. The user ID identifies the user when accessing the system. All files and job variables are set up under a user ID. The names of the files and job variables are stored in the *file catalog* together with the user ID.

user privileges

All the attributes that represent rights that are assigned to a *user identification* in a BS2000 system and are stored in the *joinfile*.

variable length record

A record in a file all of whose records may be of different lengths. The record length must either be specified in a record length field at the start of the record or must be implicitly distinguishable from the next record through the use of a separator (e.g. Carriage Return - Line Feed).

virtual filestore

The FTAM virtual filestore is used by *FT systems* acting as *responders* to make their files available to their *partner systems*. The way a file is represented in the virtual filestore is defined in the FTAM standard, see *file attributes*.

VisibleString

Character repertoire for files transferred to and from *FTAM partners*.

volume set

Component of an SM pubset in a BS2000 system. A volume set is a set of disks whose key properties (disk format, allocation unit) match.
The name of the volume set is administered in a directory of the SM pubset.
However, the data on a volume in the volume set is addressed via the SM pubset ID.

VSAM

IBM file access method for sequential, direct and indexed access.

VTAM

IBM telecommunication access method.

WAN (Wide Area Network)

A public or private network that can span large distances but which runs relatively slowly and with higher error rates when compared to a *LAN*.
Nowadays, these definitions have only limited validity. Example: in ATM networks.

X.25

X.25 is a standard protocol suite for packet switched wide area network (WAN) communication. It is designed as three conceptual layers, which correspond closely to the lower three layers of the seven-layer OSI model.

X terminal

In Unix systems: A terminal or software component to display the graphical X Window interface of Unix systems. An X terminal or a corresponding software emulation is a prerequisite for using the graphical interface of openFT.

10 Abbreviations

ACSE	Association Control Service Element
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
API/CS	Application Programming Interface/Communication System
APPC	Advanced Program-to-Program Communication
APPN	Advanced Peer-to-Peer Networking
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
ASECO	Advanced Security Control
ASN	Abstract Syntax Notation
ATM	Asynchronous Transfer Mode
BCAM	Basic Communication Access Method
BSI*	Bundesamt für Sicherheit in der Informationstechnik (German Federal Office for Information Security)
CAE	Common Application Environment
CCP	Communication Control Program
CCS	Coded Character Set
CCSN	Coded Character Set Name
CDDI	Copper Distributed Data Interface
CEN	Comité Européen de Normalisation (European Committee for Standardization)
CENELEC	Comité Européen de Normalisation Électrotechnique (European Committee for Electrotechnical Standardization)
CICS	Customer Information Control System (IBM)
CMX	Communication Manager Unix Systems
COM	Communication Port (asynchronous)
CPX	Compact Packet Exchange

CSV	Character Separated Values
DAS	Data Access Service
DAP	Directory Access Protocol
DBA	Data Base Access Service
DCAM	Data Communication Access Method
DCE	Data Communication Equipment
DCE	Distributed Computing Environment (OSF)
DCM	Data Communication Method
DES	Data Encryption Standard
DFR	Document File Retrieval
DFS	Distributed File System (DCE)
DIN*	Deutsches Institut für Normung (German standards institute)
DME	Distributed Management Environment
DMS	Data Management System
DNS	Domain Name Service
DOS	Disk Operating System
DSA	Directory System Agent
DSC	Data Stream Compatibility
DSM	Distributed Systems Management
DSP	Directory System Protocol
DSSM	Dynamic Subsystem Management
DTE	Data Termination Equipment
DTS	Distributed Time Service
EBCDIC	Extended Binary-Coded Decimal Interchange Code
EN	European Norm
ENV	Europäischer Normen-Vorschlag * (European prestandard)
EPHOS	European Procurement Handbook for Open Systems
ERMS	Entity Relationship Management System

ES	End System
ETSI	European Telecommunication Standards Institute
EWOS	European Workshop for Open Systems
FADU	File Access Data Unit
FDDI	Fiber Distributed Data Interface
FEP	Front End Processor
FJAM	File Job Access Method
FMLI	Form and Menu Language Interpreter
FSB	Forwarding Support Information Base
FSS	Forwarding Support Service
FT	File Transfer
FTAC	File Transfer Access Control
FTAM	File Transfer, Access and Management (ISO 8571)
FTP	File Transfer Protocol
FTPS	FTP via SSL / TLS
GOSIP	Government OSI Profile
GPL	Gnu Public License
GSM	Global System for Mobile Communication
HDLC	High Level Data Link Control (ISO 7776)
HNC	Highspeed Net Connect
HPFS	High Performance File System
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines Corporation
ICC	Intelligent Communication Controller
ICMP	Internet Control Message Protokoll
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
IGMP	Internet Group Management Protocol

IMS	Information Management System (IBM)
IP	Internet Protocol
ISAM	Index Sequential Access Method
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
IT	Information Technology
ITSEC	Information Technology Security Evaluation Criteria (Europe, White Book)
ITU	International Telecommunication Union
JCL	Job Control Language
LAN	Local Area Network
LMS	Library Maintenance System
LU	Logical Unit
MAC	Medium Access Control
MAN	Metropolitan Area Network
MCR	Magnetic Card Reader
MIB	Management Information Base
MLC	Modular LAN Connect
MSV*	Mittelschnelles Synchron Verfahren (Medium-fast synchronous method)
MVS	Multiple Virtual System
NCP	Network Control Program (SNA)
NCS	Network Control System
NDMS	Network Data Management System
NEA	(Name der TRANSDATA-Architektur von Siemens)
NFS	Network File System
NIS	Network Information Service
NTP	Network Time Protocol
ODI	Open Data Link Interface
ODI	Open Device Interface

ODL	Object Description Language
OSI	Open Systems Interconnection
OSS	OSI Session Service
PAM	Pluggable Authentication Modules
PC	Personal Computer
PCMX	Portable Communication Manager Unix Systems
PDU	Protocol Data Unit
PEM	Privacy Enhanced Mail
PICS	Protocol Implementation Conformance Statement
PIN	Personal Identification Number
PKCS	Public Key Cryptography Standards
PLAM	Primary Library Access Method
POP	Post Office Protocol
POSIX	Portable Operating System Interface for Open Systems
PSDN	Packet Switched Data Network
PU	Physical Unit
RFC	Request for Comments
RFC1006	Request for Comments 1006
RJE	Remote Job Entry
RPC	Remote Procedure Call
RMS	Reliant Monitor Services
RTS	Reliable Transfer Service
SAM	Sequential Access Method
SAP	Server Advertising Protocol (NetWare)
SAP	Service Access Point (OSI)
SCM	Software Configuration Management
SDF	System Dialog Facility
SDLC	Synchronous Data Link Control

SESAM*	System zur Elektronischen Speicherung Alphanumerischer Merkmale (Database System running on BS2000 Systems)
SMF	Service Management Facility (Solaris)
SMF	System Management Facility (IBM)
SMTP	Simple Mail Transfer Protocol
SNA	Systems Network Architecture
SNMP	Simple Network Management Protocol
SNPA	Subnetwork Point of Attachment
SQL	Structured Query Language
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TELNET	Telecommunications Network Protocol
TFTP	Trivial File Transfer Protocol
TID	Transport Identification
TLS	Transport Layer Security
TNSX	Transport Name Service in Unix systems
TPI	Transport Protocol Identifier
TS	Transport System
UDP	User Datagram Protocol
UDS	Universal Database System
URL	Uniform Resource Locator
UTM	Universal Transaction Monitor
VDE*	Verband deutscher Elektrotechniker (German electrical engineers association)
WAN	Wide Area Network
WS	Workstation
XDR	External Data Representation
XDS	API to Directory Service

* German abbreviation

11 Related publications

You will find the manuals on the internet at http://bs2manuals.ts.fujitsu.com. You can order printed copies of those manuals which are displayed with an order number.

openFT documentation

openFT
Concepts and Functions
User Guide

openFT (BS2000)
Installation and Operation
System Administration Guide

openFT (BS2000)
Command Interface
User Guide

openFT (BS2000)
Program Interfaces
Programmer Reference Guide

openFT (Unix and Windows Systems)
Installation and Operation
System Administration Guide

openFT (Unix and Windows Systems)
Command Interface
User Guide

openFT (Unix and Windows Systems)
C and Java Program Interface

Programmer Reference Guide

openFT (Unix and Windows Systems)
openFT-Script Interface
Programmer Reference Guide

openFT (z/OS)
Installation and Operation
System Administration Guide

openFT (z/OS)
Command Interface
User Guide

Documentation for the BS2000 environment

openNet Server (BS2000)
BCAM
User Guide

SNMP Management
SNMP Management for BS2000/OSD
User Guide

BS2000 OSD/BC
Commands (multiple volumes)
User Guide

BS2000 OSD/BC
Executive Macros
User Guide

IMON (BS2000)
Installation Monitor
User Guide

BS2000 OSD/BC
Introduction to DMS
User Guide

BS2000/OSD-BC
Subsystem Management (DSSM/SSCM)
User Guide

BS2000 OSD/BC
System Installation
User Guide

BS2000 OSD/BC
Introduction to System Administration
User Guide

JV (BS2000)
Job Variables
User Guide

SECOS (BS2000)
Security Control System
User Guide

XHCS (BS2000)
8-bit-Code- and Unicode Support in BS2000/OSD
User Guide

RAV (BS2000)
Computing Center Accounting Procedure
User Guide

HIPLEX AF (BS2000)
High Availability Applications in BS2000/OSD
User Guide

Documentation for the Unix system environment

CMX
Operation and Administration
User Guide

CMX
Porgramming Applications
Programming Manual

OSS(SINIX)
OSI Session Service
User’s Guide