



bTrade TDAccess

OS/390 User Guide

Release 2.2

© 2003 bTrade
All rights reserved.
Document 2003.UG020.01
Printed in the USA.

CONFIDENTIAL. All rights reserved. This document, including any writing, drawings, notes, or verbal representation made or shown in the course of this communication is confidential and proprietary to bTrade. No part of the materials included in this communication should be: a) reproduced; b) published in any form by any means, electronic or mechanical, including photocopy or information storage or retrieval system; or c) disclosed to third parties, without the express written authorization of bTrade.

Contents

Contents	3
1 Purpose.....	5
2 Introduction.....	6
2.1 Overview.....	6
2.2 Using the AS1 Network.....	7
2.3 Using the AS2 Network.....	7
2.4 Environments	7
2.4.1 Networks	8
2.4.2 Components	8
3 System Requirements.....	9
4 Installing TDAccess for OS/390.....	10
4.1 Sample Installation.....	10
5 Configuring TDServer	13
5.1 Customizable Parameters.....	13
6 Communication with TDNgine Hubs	20
6.1 IMPORT Utility	21
6.2 Utilizing TDClient to Execute Send/Receive Transfers	25
6.2.1 Sample Send Transfer:	25
6.2.2 CHGPASSWD.....	26
6.2.3 Sample CPLOOKUP.TBL.....	28
6.2.4 Execution of a Secured Non-EDI send transfer using SECFILEX:	28
6.2.5 Executing a Secured Send transfer using Instream CMDFILE parameter	30
6.2.6 Executing a Receive transfer using the CMDFILE	31
6.2.7 Receiving a FILE:	33
7 Communication with FedExNet	34
7.1 IMPORT utility	34
7.2 Utilizing TDClient to Execute Send/Receive Transfers	37
7.2.1 Sample BATCH job executing a send transfer pointing to a command parameter PDS member:	37
7.2.2 Sample BATCH job executing a send transfer using the Instream CMDFILE parameter	39
7.2.3 Sample BATCH job executing a receive transfer using the CMDFILE...	40
7.2.4 Receiving a file	42
7.2.5 Example BATCH job executing a receive transfer using the Instream CMDFILE	42
8 Communication with IBM/IGN	44
8.1 Certificates	44
8.2 CMDPARSE utility	44
8.3 Utilizing TDClient to Execute Send/Receive Transfers	46
8.3.1 Example BATCH job executing a send transfer pointing to a command parameter PDS member	47

8.3.2	Example BATCH job executing a send transfer using the Instream CMDFILE parameter.....	48
8.3.3	Example BATCH job executing a receive transfer using the CMDFILE	50
8.3.4	Receiving a FILE:	51
8.3.5	Example BATCH job executing a receive transfer using the Instream CMDFILE	51

1 Purpose

The purpose of this document is to aid in the installation and implementation of TDAccess for OS/390. Refer to the *bTrade TDAccess CLI and API User Guide* for detailed instructions on using the command-line interface.

2 Introduction

2.1 Overview

Welcome to TDAccess, EDI data communications software from bTrade. TDAccess has a simple-to-install, easy-to-use graphical user interface (GUI) for sending and receiving both EDI and non-EDI data files electronically via FTP, SMTP/POP3, ASI (EDI-INT), AS2 (HTTP with S/MIME). For the Unix, MVS and AS/400 computing platforms, TDAccess provides an easy-to-use command-line interface (CLI). The command-line interface can be used on all computing platforms and is documented in the *TDAccess CLI and API User Guide*. TDAccess resolves transport commands, site EXECS, and remote procedure calls common to the host server implementation(s). The user specifies the mailbox (user ID) and the file(s) to be sent or received and TDAccess does the rest.

TDAccess was specifically developed to satisfy the overwhelming demand for secure TCP/IP-based FTP, e-mail, and HTTP file transfers over internet, intranet, and extranet environments, and is ideal for both client and workstation platforms. RSA Public Cryptography, MD5 Authentication, and DES Encryption functionality is provided, that adheres to the ANSI X12.58 standards established by the Data Interchange Standards Association (DISA). Additionally, TDAccess generates X.509 certificate requests, which can be validated by a private Certifying Authority (TDManger) or resolved via TDManger with a public Certifying Authority, such as Entrust Technologies.

TDAccess supports SMTP/POP3, and a variety of FTP implementations including:

- FedEx Net® (FEDEXNET and FEDEXNET X12.58)
- AT&T® Global Network FTP interface to the IBM Information Exchange (IBM-IE)
- Wal-Mart (WALMART)
- Sterling Commerce (STERLING-COMMERCE, Dataguard and CONNECTMAIL)
- GEIS (EDI*EXPRESS, Enterprise EDI and EDISWITCH)
- MCI (MCI-EDI*NET)
- bTrade's FTP EDI servers (EAFTP)
- Generic FTP servers (GENERIC-MVS, GENERIC-DOS, and GENERIC-SSL)
- E-mail servers (SMTP/POP3)

Several other custom FTP interfaces are available upon request. Additionally, TDAccess supports AS1 (EDI-INT), AS2 (HTTP with S/MIME).

TDAccess facilitates pre-configuration using runtime files (tdclient.ini and exfer.ini) and software distribution via the World Wide Web. Pre-configuration, which enables the user to see the values appropriate for them, may be performed prior to shipping the software to them. For example, if a user wanted to send a file to IBM's Information Exchange

service, the login may be pre-configured with the correct login name, password and IP addresses for that network.

2.2 Using the AS1 Network

The AS1 network style allows for secure communications over email using S/MIME. A centralized Trading Partner Hub usually distributes keys, certificates and configuration files.

Requirements:

- POP3 Server IP Address, User ID and Password
- SMTP Gateway IP Address
- EDI Name

Trading Partners must be defined in the Trading Partner Address Book in order for the TDAccess Client to communicate with them. For more information, refer to the Trading Partner Address Book section in this document.

2.3 Using the AS2 Network

The AS2 network style allows for secure communications over HTTP using SSL (HTTP/S) with S/MIME. Keys, certificates and configuration files are usually distributed by a centralized Trading Partner Hub.

After a TDAccess client receives the certificates as downloaded from TDManager; the user can send and receive secure files to and from its trading partners via the Internet, intranet(s) and extranet(s).

For information on World Wide Web distribution, Private Certifying Authority (TDManager), TDAccess and Enabling Services available from bTrade, please call 1-877-4BTRADE or visit our website at www.bTrade.com.

2.4 Environments

A wide variety of operating environments and implementations are available from bTrade, allowing our clients to use TDAccess as their enterprise-wide solution for reducing network costs and securing data.

TDAccess 2.x is available for the following environments: (Includes AS1/AS2 communications)

- IBM or Compatible PCs: Windows 98 Second Edition, Windows NT and Windows 2000
- HP-UX11
- Sun Solaris
- AIX
- Linux

- MVS
- AS/400
- Contact your bTrade sales representative for other platforms

2.4.1 Networks

TDAccess is designed for TCP/IP-based networks and uses FTP, HTTP/HTTPS or SMTP/POP3 for file transfers.

2.4.2 Components

TDAccess contains the following components:

- TDComm-Press Extended Security Option: The data may be DES encrypted, MD5 or SHA1 authenticated and RSA® digitally signed as well as compressed. It uses X12.58 data security structures and is based on TDManager selected options. This product is built-in to TDAccess and is no longer available stand-alone.
- SSL 3.0: Employs the Secure Socket Layer (SSL) version 3.0 from Netscape when negotiating the FTP logon and uses TDComm-Press Extended Security Option to compress the data (COMPRESS=Y in the tdclient.ini file). From logon to session end, the data and command sockets are encrypted using DES encryption generated by BSAFE libraries from RSA.
- AS2: (TDAccess 2.x ONLY) Built-in HTTP/S server for secure communications using S/MIME and SSL over HTTP for AS2.
- AS1: Secure email using S/MIME. Both AS1 and AS2 network styles utilize the functions and features of securing data integrated within TDManager.

3 System Requirements

The software does not include the TCP/IP libraries and assumes that the TCP/IP connection is available via a dedicated circuit or dial-up.

TDClient has been tested and used with the following:

- With dedicated TCP/IP access under:
- Windows 95/98/NT/2000
- Solaris 2.6
- OS/390 2.4
- OS/400
- HP-UX 11.0
- AIX 4.2
- Dial access using Dial-Up Networking under Windows 95/98/NT/2000.

TDClient for OS390 is intended to run in batch mode on OS390 2.4 or greater. The operating system must be running Open/Edition and be POSIX compliant.

4 Installing TDAccess for OS/390

The executable file must be saved to a folder on a Window's PC for execution. If one of the referenced files below does not exist on the CD, please contact Customer Support. Once the executable file has been saved to a temp folder on the PC it will be necessary to double-click on it to execute. The executable file will expand out into three files:

- decomp.log
- README.txt
- XMIT.BIN

The README.txt contains a quick review of the installation process. This document is intended to elaborate on the installation and provide instructions for using the product.

The XMIT.BIN is a binary file which contains the actual software.

The CD will consist of an executable file, a customized configuration file (easyacc.ini or tdclient.ini – depending on which version of SecureManager or TDManger was used to generate the configuration and runtime files), and a runtime key file (optional).

- EAMVS<version>.exe
- easyacc.ini
- runtime.rtm

In the following steps, replace "xmit.dataset" and "install.dataset" with the dataset names appropriate for the installation.

1. FTP the accompanying file (XMIT.BIN) in BINARY mode to an **MVS Dataset** allocated with the following attributes:

```
//XMITFILE DD DSN=xmit.dataset,DISP=(,CATLG,DELETE),
//      VOL=SER=volser,UNIT=3390,SPACE=(CYL,(15,5)),
//      DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120)
```

1. Using TSO, issue the following RECEIVE command:
RECEIVE INDA('xmit.dataset')
2. To the Receive Parameters prompt, enter:
DA('install.dataset')
3. Edit the install.dataset file. Read the \$README file there for further instructions to complete the installation.

The above instructions will create an Installation Library containing the files required to install the TDAccess for MVS product.

4.1 Sample Installation

Edit and run the install job from PDS member \$INSTALL

```
//jobname JOB (acct),pgmr,MSGLEVEL=1,REGION=7M,CLASS=A,
// MSGCLASS=X,NOTIFY=user
//*
//* MEMBER $INSTALL
//*
//* TDClient V1 MVS Installation JCL.
//*
//* Make the following changes:
//*
//* 1) Provide the appropriate fields on the JOBCARD, above.
//* 2) Change all occurrences of install.dataset to the name
//*     of this dataset.
//* 3) Change all occurrences of user.tdload to a valid destination
//*     dataset name.
//* 4) Change all occurrences of user.tdsamp to a valid destination
//*     dataset name.
//* 5) Change all occurrences of user.cpsamp to a valid destination
//*     dataset name.
//*
//*****TSO Receive for DISTLIB and SAMPLIB Datasets.*****
//*****TSO Receive for DISTLIB and SAMPLIB Datasets.*****
//*****
//RECEIVE EXEC PGM=IKJEFT01,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//TDLOAD DD DSN=install.dataset(TDLOAD),DISP=SHR
//TDSAMP DD DSN=install.dataset(TDSAMP),DISP=SHR
//TDSSAMP DD DSN=install.dataset(TDSSAMP),DISP=SHR
//CPSAMP DD DSN=install.dataset(CPSAMP),DISP=SHR
//SYSTSIN DD *
RECEIVE INFILE(TDLOAD)
DATASET('user.tdload')
RECEIVE INFILE(TDSAMP)
DATASET('user.tdsamp')
RECEIVE INFILE(TDSSAMP)
DATASET('user.tdssamp')
RECEIVE INFILE(CPSAMP)
DATASET('user.cpsamp')
/*
//
```

The above job will allocate three PDS datasets: TDLOAD, TDSAMP, and CPSAMP.

- TDLOAD - TDClient Loadlib containing the executable Load Modules.

-
- TDSAMP - TDClient sample JCL for running TDClient Batch jobs. Also, includes JCL for generating, importing, and parsing of keys. Appropriate JCL to use is dependent on the Hub administering the keys.
 - Batch - Sample JCL for executing a send and receive transfer.
 - Clist - Sample Clist for executing a send and receive transfer.
 - CmdParse - For IBM Users Only. Parses Key file received from IBM.
 - GenKeys - Generates RSA Key Pairs.
 - Import - Import utility for PKI keys received from Key Administration.
 - CPSAMP - TDCompress sample JCL, Clist, and API's for executing TDCompress.

Pre-allocate two datasets to facilitate the customer configuration data and the stored transfer repository.

Note:

The TDClient EASYACC dataset is used to hold the customer information and network configuration. This dataset is referenced by the DD EASYACC.

Allocate the TDClient EASYACC dataset with the following attributes:
Recfm=FB, lrecl=200, blksize=27920

The TDClient EXFER dataset is used to hold newly created stored transfer. This dataset is reference by the DD EXFER.

Allocate the TDClient EXFER dataset with the following attributes:
Recfm=FB, lrecl=80, blksize=27920

FTP the TDAccess *easyacc.ini* or *tdclient.ini* file, received during download, into the pre-allocated TDClient EASYACC dataset in ASCII mode. The TDClient EXFER dataset can remain empty. However, it must be referenced in the BATCH job located in the TDSAMP PDS dataset. TDClient will write the transfer to the TDClient EXFER dataset during execution.

5 Configuring TDServer

To setup and customize TDServer to run in your environment, the tdserver.cfg file that is created on the Unix portion of the MVS environment needs to have its parameters set to be in sync with the environment. There are two ways this can be accomplished:

1. Editing the sample hlq.TDSSAMP(TDSERVER) file before extracting the configuration file into the HFS (Unix) file system, or
2. Directly editing the tdserver.cfg file after it is imported into the HFS (Unix) file system (directly under the TDServer directory).

5.1 Customizable Parameters

The following table details the parameters to customize for your environment.

TDServer Configuration

Log Options		
Parameter	Recommended Value	Description
consoleLog	Y (unless running as a service)	(Y or N) Enable/disable logging display to console terminal
logDataLimit	<i>No longer used</i>	(Positive integer) Max size of comm log-file
commLogLevel	6	(-6 to 6) General communications log-file logging level
SMIMELogLevel	0	(-6 to 6) S/MIME logging level The commLogLevel and SMIMELogLevel use the following scale: 0 => No logging; 1 => minimal logging; 6 => maximum logging; -1, -2, ..., -6 => same as positive value except each time an entry is written, the log-file is closed and re-opened. The negative values are only for trouble shooting with Tech Support.
socketLogLevel	0	(0 to 4) SSL logging level 0 => No logging; 1 => minimal logging; 4 = maximum logging. Maximum logging should only be used when trouble shooting with Tech Support.
logDir	/TDServer/logs	(Text) Directory for comm and SSL logs, if enable

keepTemp	0	(1 or 0) Keep temp files (1) or delete when finished (0) As each message is processed, a temporary directory is created in the directory specified by 'tmpDir' (see below) and a number of temporary files are generated within this temp directory. If no processing errors occur, then all temp files and the temp directory are deleted when 'keepTemp' is 1. However, if a processing error occurs, then the temp files and temp sub-directory are kept regardless of the keepTemp value.
Listen Ports		
httpPort	8001	(1-65536) Port to receive HTTP messages
httpsPort	8002	(1-65536) Port to receive HTTPS messages
idleTimeout	900	(Positive integer; default value = 900) Timeout, in seconds, used in two ways: (1) Timeout expires if the client connects to the server but fails to start sending the message within 'idleTimeout' seconds of the connection. (2) Timeout expires if, after the message has been processed and a response sent to the client, the client does not start sending another message or disconnect from the server. If the 'idleTimeout' timeout expires, the server disconnects. The default value of 900 seconds (15 minutes) is used if no value is specified in the tdserver configuration file.
receiveTimeout	60	(Positive integer; default value = 60) Timeout, in seconds, used in two ways: (1) During SSL negotiation, while waiting for response from the server (there are several 'steps' in the negotiation where the client waits to receive the server's response). (2) After message transmission has been started by the client, the client may pause, or transmission may be paused by network congestion; the 'receiveTimeout' value is the longest acceptable pause during message transmission. If the 'receiveTimeout' timeout expires, the server disconnects. The default value of 60 seconds (1 minute) is used if no value is specified in the tdserver configuration file.

sendTimeout	60	(Positive integer; default value = 60) Timeout, in seconds, while sending the server's response to the client. If the 'sendTimeout' timeout expires, the server disconnects. The default value of 60 seconds (1 minute) is used if no value is specified in the tdserver configuration file.
httpUsers	32	(Positive integer, or zero) When non-zero, specifies the maximum number of concurrent HTTP sessions. A value of zero disables all HTTP sessions. The default value of 32 (concurrent sessions) is used if no value is specified in the tdserver configuration file.
httpsUsers	32	(Positive integer, or zero) When non-zero, specifies the maximum number of concurrent HTTPS sessions. A value of zero disables all HTTPS sessions. The default value of 32 (concurrent sessions) is used if no value is specified in the tdserver configuration file.
Server Parameters		
ediName	<customer EDI name>	(Text) The EDI Name of the server. Defines the single personality specified by this identifier. All incoming messages which are encrypted must be encrypted for this identifier.
as2Name	<customer AS2 name>	(Text) The AS2 Name of the server. This AS2 Name appears in all MDNs sent back to clients.
emailAddress	<customer server administrator's email address>	(Text) The EMail Address of the Server Administrator. This EMail Address appears in all MDNs sent back to clients. Note the AS2 server will never send an EMail to this address.
SSL Options		
authenticateClients	0	(0 or 1) Specifies if the server will (1) or will not (0) require that HTTPS clients attempting to connect authenticate themselves during the SSL negotiation by sending a copy of their public certificate to the TDServer for comparison with the local copy.
runtimeDir	/TDClient/Runtime	(Text) Directory containing runtime database files, cert.fil and private.fil, which are used in lieu of a TDManger database.

certDir	/TDServer/certs	(Text) Directory into which received certificates are placed. The TDServer will receive certificates via PKCS7 'certs-only' messages, and will deposit the recovered (public) certs into this directory for manual review and manual inclusion into the TDManager database. (See TDManager documentation for how import and export participant certificates).
Incoming Message Processing Parameters		
ediDir	/TDServer/edi	(Text) Directory for storing all successfully received data. If 'keepEncryptedOnly' is 0, then the data payload(s) for each message are stored here after successful message decryption and authentication. If 'keepEncryptedOnly' is 1, then only the original (encrypted) message files are placed here and the decrypted data (payload) files are deleted.
mdnDir	/TDServer/mdns	(Text) Directory for storing received Message Disposition Notifications (MDNs). Omit this line (or comment it out) if no MDNs are to be saved to disk. See also dbDsn at al below.
tmpDir	/TDServer/Temp	(Text) Directory used for storing temporary files and temporary directories.
errorDir	/TDServer/Error	(Text) Directory to store message files which failed to be decrypted, authenticated, decompressed, etc.
TDNgineDataDir	/TDNgine/data	(Text) (Applicable only if TDServer is installed as a component of a TDNgine installation). TDNgine directory to receive incoming message files.
keepEncryptedOnly	N (Y if running software in DMZ so to not expose decrypted data)	(Y or N) Specifies if (N) decrypted/authenticated/decompressed payloads are to be stored in the 'ediDir', or if (Y) only the original (encrypted-signed/compressed) message files are to be stored in the 'ediDir'.
divertDuplicateData	Y	(Y or N) Specifies if (N) duplicate messages are to be processed normally, with the recovered data payloads moved to 'ediDir', or if (Y) duplicate messages should be moved to 'errorDir'. A duplicate message is a message containing either the identical data payload, or the identical message ID, as a previously received message. NOTE: This feature is only available if an MDN database is being used.

securityLevelRequired	PLAIN_DATA	("PLAIN_DATA", "SIGNED_DATA", "ENCRYPTED_DATA", or "SIGNED_ENCRYPTED_DATA") Specifies the minimum level of security acceptable for incoming messages. "PLAIN_DATA" - no security is required and any message will be accepted for processing. "SIGNED_DATA" - an incoming message must be at least signed by the sender for it to be processed. Unsigned messages will be rejected and moved to 'errorDir'. "ENCRYPTED_DATA" - an incoming message must be encrypted for the server for it to be processed. Unencrypted messages will be rejected and moved to 'errorDir'. "SIGNED_ENCRYPTED_DATA" - an incoming message must be both signed by the sender and encrypted for the server for it to be processed. Unencrypted/unsigned messages will be rejected and moved to 'errorDir'.
retainOriginalFileNames	Y	(Y or N) Specifies if a message's payload should be named using its original filename, if present. Some vendor's AS2 messages include the original filename, some do not. (bTrade's does). When set to 'Y', the autoExtLen value is used to assign unique filenames in the case of where more than one message uses the same original filename.
autoExtLen	3	(0-10) Used only when retainOriginalFileName=Y. Specifies the number of characters available in a numeric extension made to a payload's filename when multiple payloads use the same original filename. For example, if payload file mydata.txt has been received and another message arrives with the same original filename, then its payload will be named mydata.002.txt if autoExtLen=3, or mydata.00002.txt if autoExtLen=5.
Database Parameters		
dbDsn	<customer db>	(Text) Data Source Name of an MDN database, a TDManager database, or a TDNgine database.

dbSchema	<customer schema>	(Text) Schema name for an MDN database, a TDManager database, or a TDNgine database.
dbUid	<customer uid>	(Text) UserID for an MDN database, a TDManager database, or a TDNgine database.
dbPwd	<customer pwd>	(Text) Password for an MDN database, a TDManager database, or a TDNgine database.
dbType	<type of db used>	(Text) Type of database being referenced. Allowed values include: Oracle - database is an Oracle database. DB2 - database is a DB2 database. ODBC - database is another type, being access via ODBC.
useMDNDb	Y (if database is set up, N otherwise)	(Y or N) Specifies if the designated database is to be used to store information about sent/received MDNs.
useRuntimesDb	Y (only used if TDManager from TDPeer is installed)	(Y or N) Specifies if the designated database is to be used for accessing certificate information (as opposed to static runtimes files, 'cert.fil' and 'private.fil').
Async MDN Processing Parameters		
sleepTime	60	(Positive integer) Wait time in seconds between cycles of processing asynchronous MDNs.
maxRetries	60	(Positive integer) Number of attempts to send an asynchronous MDN.
smtpIp	<customer smtp domain name or ip address>	(Text) Address of SMTP mail server for outbound email style MDNs.
asyncMdnDir	/TDServer/mdns	(Text) Directory to store async MDNs.
asyncMdnListFileSpec	MDNListFile	(Text) Name of file that queues pending MDNs to process.
asyncCommLogFileSpec	MDNLogFile	(Text) Name of file that logs MDNs processed.

asyncCommLogLevel	6	(-6 to 6) Log-file logging level for general communications log for asynchronous MDN transmissions. The following scale is used: 0 => No logging 1 => minimal logging 6 => maximum logging -1, -2, ..., -6 => same as positive value except each time an entry is written, the log-file is closed and re-opened. The negative values should only be used when troubleshooting with Tech Support.
asyncSMIMELogLevel	6	(-6 to 6) S/MIME logging level. The following scale is used: 0 => No logging 1 => minimal logging 6 => maximum logging -1, -2, ..., -6 => same as positive value except each time an entry is written, the log-file is closed and re-opened. The negative values should only be used when troubleshooting with Tech Support.
asyncSocketLogLevel	0	(0 to 4) SSL logging level 0 => No logging 1 => minimal logging 4 => maximum logging Beware, maximum logging for socketLogLevel produces a HUGE amount of logging ...
connectTimeout	60	Timeout in seconds when connecting to the server to which the MDN is to be sent.
eaSendTimeout	60	Timeout in seconds used when sending the MDN and waiting for the server response.
responseFileSpec	MDNResponseFile	File used to store the async MDN's server's response.
Firewall and Proxy Configuration		
lowClientPort	0	Numeric value > 1025. See highClientPort for description.

highClientPort	0 (typically don't want to restrict)	Numeric value > 1025. Together with lowClientPort specifies a range of client ports to be used. If lowClientPort = 0, then any available client port will be used; Else if highClientPort = 0, or lowClientPort = highClientPort, then only the port specified by lowClientPort will be used. Else, a port in the range from lowClientPort to highClientPort (inclusive) will be used. If no port is available in the specified range, then a failure is reported and the communications session is ended.
localHostIp	<customer ip>	The IP address of the machine on which the server is running. Useful in situations involving Static NAT mapping to multiply-hosted machines.
firewallHostIp	<customer ip>	IP Address of a firewall or proxy server.
firewallUserId	<customer user id>	User ID to log onto a firewall or proxy server.
firewallPasswd	<customer pwd>	Password to log onto a firewall or proxy server.
firewallType	<type of firewall used>	Numeric value which specifies what the firewall/proxy type. Allowed values include: 0 - no firewall or proxy server 2 - Proxy server 3 - static NAT'ed firewall
firewallPort	<customer port>	Port to be used when logging onto a firewall or proxy server
HFS Temp File Directory		
hfsDir	<hfs location on MVS>	The Hierarchical File System location for Unix on the MVS machine where temporary files will be located.

6 Communication with TDNgine Hubs

IMPORT keys received from your Key Administrator. Utilize the sample member IMPORT located in the TDSAMP library created during install.

Note:

An .rtm file and approval code should be provided from the HUB Key Administrator or Security Officer. If one or the other does not exist, please contact your Hub Trading Partner for additional directions.

FTP the .rtm file to the mainframe using binary format. Use the .rtm file as input to the IMPORT job. Let FTP allocate the file attributes.

Example:

```
ftp> open os390
Connected to os390.
220-FTPD1 IBM FTP CS V2R8 at S390, 18:42:56 on 2001-02-14.
220 Connection will close if idle for more than 5 minutes.
User (p390mvs:(none)): uid
331 Send password please.
Password:
230 USERID is logged on. Working directory is "UID.".
ftp> binary
200 Representation type is Image
ftp> put input.rtm 'data.set.rtmfile' rep
200 Port request OK.
125 Storing data set DATA.SET.RTMFILE
250 Transfer completed successfully.
ftp: 5248 bytes sent in 0.02Seconds 328.00Kbytes/sec.
ftp>quit
```

6.1 IMPORT Utility

(TDSAMP member (IMPORT))

STEP1: Run the Import utility – What needs to be added follows.

Note:

If the GENKEY utility was not executed, comment out the PRIVKEY DD and its references.

```
//IMPORT    JOB , 'PGMRNAME' , CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//*****
//* INSTALL THE SECURITY RUNTIME FILES. REFER TO COMMENTS THROUGHOUT
//* THIS PROCEDURE FOR INSTRUCTIONS.
//*****
//*
//IMPORT    PROC RTMFILE=,   <== DO NOT CHANGE HERE.
//           HIGHQUAL=,      PROVIDE VALUES ON THE 'EXEC'
//           APPCODE=,       STATEMENT AT THE END OF THE JCL
//           PRIVKEY=,
```

```
// STEPLIB=
//*
//***** FIRST STEP DELETES ANY EXISTING SECURITY FILES.
//*****
//*
//DELETE EXEC PGM=IEFBR14
//CERT DD DSN=&HIGHQUAL..CERT.FIL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//PRIVATE DD DSN=&HIGHQUAL..PRIVATE.FIL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//SYMKEY DD DSN=&HIGHQUAL..SYMKEY.FIL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//ALIAS DD DSN=&HIGHQUAL..ALIAS.TBL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//CPLOOKUP DD DSN=&HIGHQUAL..CPLOOKUP.TBL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//PARTIC DD DSN=&HIGHQUAL..PARTIC.TBL,DISP=(MOD,DELETE,DELETE),
//          SPACE=(TRK,(1))
//*
//***** SECOND STEP DYNAMICALLY ALLOCATES AND DECOMpresses THE
//* RUNTIME FILES FROM TDManager.  THE 'RTMFILE', CREATED
//* BY TDManager, MUST HAVE BEEN RECEIVED USING A BINARY FILE
//* TRANSFER MECHANISM.  THE 'APPCODE' IS REQUIRED IF THE RTMFILE
//* IS ENCRYPTED WITH THE PARTICIPANT'S TDManager APPROVAL CODE.
//*
//* THE SECURITY FILES CREATED BY THIS STEP ARE:
//*      CERT.FIL
//*      PRIVATE.FIL
//*      SYMKEY.FIL
//*      ALIAS.TBL
//*      CPLOOKUP.TBL
//*****
//*
//DECOMP EXEC PGM=DECOMP,REGION=2M,
//        PARM='NOUNCOMP NOVERIFY KEEPSIGS KEY=&APPCODE'
//STEPLIB DD DSN=&STEPLIB,DISP=SHR
//DCMPLOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DATAIN DD DSN=&RTMFILE,DISP=SHR
//*
//***** 
//* THIRD STEP PROCESSES THE PRIVATE KEY SECURITY FILE
//* AND DYNAMICALLY ALLOCATES/WRITES THE PASSPHRASE FILES
```

```
/*
/* IF THIS STEP FAILS, THEN RERUN THE ENTIRE JOB.
//*****
/*
//IMPPRIV EXEC PGM=IMPPRIV,REGION=2M
//STEPLIB DD DSN=&STEPLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//PRIVKEYS DD DSN=&HIGHQUAL..PRIVATE.FIL,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,DISP=(NEW,DELETE),
//          UNIT=SYSDA,SPACE=(TRK,(1,1))
/*
/* THE FOLLOWING DATASET IS ONLY REQUIRED IF THE RSA KEYS
/* WERE GENERATED IN BATCH VIA THE 'GENKEYS' PROGRAM
/* OTHERWISE, COMMENT OUT THE DD STATEMENT
/*
/*PRIVKEY DD DSN=&PRIVKEY,DISP=SHR
//          PEND
/*
//*****
/* EXECUTE THE PROCEDURE TO INSTALL THE SECURITY RUNTIME FILES.
/*
/* PROVIDE THE FOLLOWING VALUES ON THE JCL BELOW TO RUN THE PROC:
/*
/*
/* 1. THE NAME OF THE COMPRESSED RUNTIME FILE FROM THE
/*      TDMANAGER.
/*
/* 2. THE HIGH-LEVEL QUALIFIER TO USE FOR NAMING THE DECOMPRESSED
/*      SECURITY FILES. YOU MUST PROVIDE THE HIGH-LEVEL QUALIFIER
/*      IN TWO PLACES -- ON THE 'HIGHQUAL' PARAMETER OF THE 'PROC'
/*      STATEMENT AND ON THE 'DSN=' KEYWORD IN THE 'DECOMP.SYSIN'
/*      STATEMENTS.
/*
/* 3. YOUR APPROVAL CODE PROVIDED BY THE HUB TRADING PARTNER.
/*
/* 4. THE NAME OF THE 'PRIVKEY' FILE CREATED BY THE 'GENKEYS'
/*      UTILITY. IF YOU DID NOT USE 'GENKEYS' TO GENERATE YOUR
/*      RSA KEY PAIR, THEN COMMENT OUT THE 'PRIVKEY' PARAMETER
/*      ON THE PROC STATEMENT AS WELL AS THE 'PRIVKEY' DD STATEMENT.
/*
/* 5. THE NAME OF THE STEPLIB WHERE THE 'DECOMP' AND 'IMPPRIV'
/*      PROGRAMS ARE INSTALLED.
//*****
/*
//STEP010 EXEC IMPORT,
//          RTMFILE=rtmfile,
//          HIGHQUAL=highqual,
//          APPCODE=0123456789ABCDEF,
/*
/*          PRIVKEY=privkey,
//          STEPLIB=TDCLIENT.LOADLIB
```

```
/*  
 /DECOMP.SYSIN DD *  
   DSN=highqual      # <== HIGH-LEVEL QUALIFIER(S) FOR SECURITY FILES  
   LRECL=84          # <== RECORD LENGTH -- DO NOT CHANGE!  
   BLKSIZE=4204       # <== BLOCK SIZE  
   RECFM=VB           # <== RECORD FORMAT -- DO NOT CHANGE!  
   TRACKS             # <== ALLOCATION UNITS  
   PRIMARY=1          # <== PRIMARY ALLOCATION  
   SECONDARY=1         # <== SECONDARY ALLOCATION
```

Installation and Key Exchange is complete.

6.2 Utilizing TDClient to Execute Send/Receive Transfers

In order to run a transfer on OS390 it is necessary to run the BATCH job found in the TDSAMP sample JCL library. Edit the Batch job accordingly to execute either the send or receive transfers. The batch job will call the TDClient parameters in three ways.

1. Using the CMDFILE parameter and pointing to a command file Dataset or PDS member.
2. Using the CMDFILE parameter and running the parameters instream.
3. Executing the transfer from the command-line executable.

Sample BATCH job executing a send transfer using the CMDFILE parameter pointing to a command parameter PDS member:

6.2.1 Sample Send Transfer:

```
//TDCLIENT JOB , 'EASYACCESS' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE DD DSN=CMDFILE.PDS(CMDSEND),DISP=SHR
//QDATA DD DSN=DATA.TO.SEND,DISP=SHR
//*
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER      DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//              LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//ALIAS       DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP  DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
```

```
//SECFILE          DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),  
//                           LRECL=256,BLKSIZE=0,RECFM=VB  
///*  
//EDIPDS          DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),  
//                           LRECL=256,BLKSIZE=0,RECFM=VB  
//SEDILOG         DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),  
//                           LRECL=256,BLKSIZE=0,RECFM=VB  
///*  
//OUTMSG          DD SYSOUT=*  
//SYSPRINT        DD SYSOUT=*  
///*  
//CPFTPLOG        DD SYSOUT=*  
//RESPLOG         DD SYSOUT=*  
//COMPLOG         DD SYSOUT=*  
//EAFTPLOG        DD SYSOUT=*           /* "LOG_FTP" IN EASYACC.INI */  
//EALOG            DD SYSOUT=*           /* "LOG_EASYACC" IN EASYACC.INI */  
//EXFERLOG        DD SYSOUT=*           /* "LOG_XFER" IN EASYACC.INI */  
//EAINILOG         DD SYSOUT=*           /* "LOG_INI" IN EASYACC.INI */  
//EAMEMLOG        DD SYSOUT=*           /* "LOG_MEM" IN EASYACC.INI */  
///*
```

Note:

When executing the first transfer it is necessary to change or alter the password.
CHGPASSWD illustrates the first transfer to be executed. This only needs to be done once.

6.2.2 CHGPASSWD

```
*****  
NETWORK="PORTALNAME"  
FTPPASSWD=oldpassword/newpassword/newpassword  
TRANSFER=(NAME=TEST  
SEND=DD:QDATA  
SENDCLASS=TEST  
SENDUSERID=USERID  
COMPRESS=Y  
CRLF=Y  
SENDASCII=Y)  
*****
```

Associated Command File:

Executing a Compressed, Non-Secured Send Transfer

SEND NON-EDI TRANSFER:

```
*****
```

```
NETWORK="PORTALNAME"
FTPPASSWD=newpassword
TRANSFER=(NAME=TEST
SEND=DD:QDATA
SENDCLASS=TEST
SENDUSERID=<to USERID>
COMPRESS=Y
CRLF=Y
SENDASCII=Y)
*****
```

Note:

Do not use the "<>". These are for documentation purposes only.

Associated Command File:
Executing a Compressed, Non-Secured, EDI Send Transfer

SEND EDI TRANSFER:

```
*****
NETWORK="PORTALNAME"
FTPPASSWD=password
TRANSFER=(NAME=TEST
SENDEDI=DD:QDATA
SENDCLASS=TEST
COMPRESS=Y
CRLF=Y
SENDASCII=Y)
*****
```

Associated Command File:
Executing a Compressed, Secured, Non-EDI Send Transfer

SEND EDI TRANSFER:

```
*****
NETWORK="PORTALNAME"
FTPPASSWD=password
TRANSFER=(NAME=TEST
SENDEDI=DD:QDATA
SENDCLASS=TEST
COMPRESS=Y
CRLF=Y
SENDASCII=Y
SECURE=Y
```

```
OTHER_COMP_PARMS='ASII CRLF SECFILE=DD:SECFILEX')
*****
```

Note:

A SECFILEX DD must be added to the JCL when Securing a Non-EDI file. The sender and receiver pairs including the transaction ID must be specified in the SECFILEX DD dataset. The parameter “OTHER_COMP_PARMS='SECFILE=DD:SECFILEX’” points to the DD containing the relationship. The SECFILEX DD is only needed for securing Non-EDI data. When securing EDI data, the sender and receiver pair will be picked up from the ISA header noted in the EDI file.

The relationship that is specified in the SECFILEX dataset must also be referenced in the CPLOOKUP.TBL (See Compress Users Guide for Securing Data).

6.2.3 Sample CPLOOKUP.TBL

```
SENDER(SENDER) RECEIVER(RECEIVER) TRANSACTION(*) ;
```

6.2.4 Execution of a Secured Non-EDI send transfer using SECFILEX:

```
//TDCLIENT JOB , 'EASYACCESS' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB   DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE   DD DSN=CMDFILE.PDS(CMDSEND),DISP=SHR
//QDATA     DD DSN=DATA.TO.SEND,DISP=SHR
//*
//EASYACC   DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER     DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//*
```

```
//ALIAS      DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP   DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
//SECFILE    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//SECFILEX   DD DSN=TDCLIEN.SECFILEX,DISP=SHR
///*
//EDIPDS     DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
///*
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
///*
//CPFTPLOG   DD SYSOUT=*
//RESPLOG    DD SYSOUT=*
//COMPLOG    DD SYSOUT=*
//EAFTPLOG   DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG       DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
///*
//*
```

NON-EDI SECURED TRANSFER:

```
*****
NETWORK="PORTALNAME"
FTPPASSWD=password
TRANSFER=(NAME=TEST
SEND=DD:QDATA
SENDCLASS=TEST
SENDUSERID=USERID
COMPRESS=Y
SECURE=Y
CRLF=Y
FILTER=Y
SENDASCII=Y
OTHER_COMP_PARMS='SECFILE=DD:SECFILEX')
*****
```

Associated Command File: Executing a Compressed, Secured, EDI Send Transfer

EDI SECURED TRANSFER:

```
*****
NETWORK="PORTALNAME"
FTPPASSWD=password
TRANSFER=(NAME=TEST
SENDEDI=DD:QDATA
SENDCLASS=TEST
COMPRESS=Y
SECURE=Y
CRLF=Y
SENDASCII=Y)
*****
```

6.2.5 Executing a Secured Send transfer using Instream CMDFILE parameter

Sample BATCH job:

```
//TDCLIENT JOB , 'EASYACCESS' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB   DD DSN=TDLOAD.LOADLIB,DISP=SHR
//*
//CMDFILE   DD *
NETWORK=PORTALNAME FTPUSERID=USERID FTPPASSWD=PASSWORD
RESET
TRANSFER=(NAME=SENDTEST SENDUSERID=USERIDSENDINGTO
SEND=DD:QDATA
SENDCLASS=TEST
COMPRESS=Y
SECURE=Y
SENDASCII=Y
FILTER=Y
OTHER_COMP_PARMS='SECFILE=DD:SECFILEX')
//*
//QDATA    DD DSN=DATA.TO.SEND,DISP=SHR
//*
//EASYACC  DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER    DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
```

```
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
///*
//ALIAS      DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP  DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
//SECFILE    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SECFILEX   DD DSN=TDCLIEN.SECFILEX,DISP=SHR
///*
//EDIPDS     DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
///*
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
///*
//CPFTPLOG   DD SYSOUT=*
//RESPLOG    DD SYSOUT=*
//COMPLOG    DD SYSOUT=*
//EAFTPLOG   DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG       DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
///*
//*
```

6.2.6 Executing a Receive transfer using the CMDFILE

Sample BATCH job:

```
//TDCLIENT JOB , 'EASYACCESS', CLASS=A, MSGCLASS=X, NOTIFY=USERID
///*
//EASYACC  EXEC PGM=TDCLIENT, REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
```

```
/*
//STEPLIB    DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE    DD DSN=CMDFILE.PDS(CMDRECV),DISP=SHR
//RECFILE    DD DSN=RECEIVE.FILE,DISP=(NEW,CATLG),
//                           UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=80,BLKSIZE=0,RECFM=FB
///*
//EASYACC    DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER      DD DSN=EXFER.INI.FILE,DISP=SHR
///*
//SYSUT1     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=8192,BLKSIZE=0,RECFM=VB
///*
//ALIAS      DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP  DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
//SECFILE    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                           LRECL=256,BLKSIZE=0,RECFM=VB
///*
//EDIPDS     DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//                           LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                           LRECL=256,BLKSIZE=0,RECFM=VB
///*
//DATAXX     DD DSN=REJECT.FILE,DISP=(NEW,CATLG),
//                           UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                           LRECL=80,BLKSIZE=0,RECFM=FB
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
///*
//CPFTPLOG   DD SYSOUT=*
//RESPLOG    DD SYSOUT=*
//COMPLLOG   DD SYSOUT=*
//EAFTPLOG   DD SYSOUT=*
//                           /* "LOG_FTP" IN EASYACC.INI */
```

```
//EALOG      DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
/*
```

6.2.7 Receiving a FILE:

RECEIVE NON-EDI TRANSFER:

```
*****
NETWORK="QRS ITX SSL"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=QDATA
RECEIVE=DD:RECVFILE
RECEIVECLASS=TEST
UNCOMP=N
CRLF=Y
AUTOEXT=N
APPEND=Y)
*****
```

RECEIVE EDI TRANSFER

```
*****
NETWORK="QRS ITX SSL"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=QDATA
RECEIVEEDI=DD:RECVDATA
RECEIVECLASS=TEST
UNCOMP=N
CRLF=Y
AUTOEXT=N
APPEND=Y)
*****
```

7 Communication with FedExNet

IMPORT keys (runtime.rtm) received on the distribution CD. Utilize the sample member IMPORT located in the TDSAMP library created during install.

FTP the *.rtm file over to the mainframe in binary format. Use the *.rtm file as input to the IMPORT job. Let FTP allocate the file attributes.

Example:

```
ftp> open os390
Connected to os390.
220-FTPD1 IBM FTP CS V2R8 at S390, 18:42:56 on 2001-02-14.
220 Connection will close if idle for more than 5 minutes.
User (p390mvs:(none)): uid
331 Send password please.
Password:
230 USERID is logged on. Working directory is "UID.".
ftp> bi
200 Representation type is Image
ftp> put input.rtm 'data.set.rtmfile' rep
200 Port request OK.
125 Storing data set DATA.SET.RTMFILE
250 Transfer completed successfully.
ftp: 5248 bytes sent in 0.02Seconds 328.00Kbytes/sec.
ftp>quit
```

7.1 IMPORT utility

(TDSAMP member (IMPORT))

The IMPORT utility will generate the KEY files necessary for encryption. The datasets generated from the IMPORT job will need to be referenced in the batch job JCL.

Note:

If you did not run the GENKEY utility, comment out the PRIVKEY DD and its references.

```
//IMPORT    JOB , 'PGMRNAME' , CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//*****INSTALLED FILES*****
///* INSTALL THE SECURITY RUNTIME FILES. REFER TO COMMENTS THROUGHOUT
```

```
/* THIS PROCEDURE FOR INSTRUCTIONS.  
*****  
/*  
//IMPORT PROC RTMFILE=, <== DO NOT CHANGE HERE.  
// HIGHQUAL=, PROVIDE VALUES ON THE 'EXEC'  
// APPCODE=, STATEMENT AT THE END OF THE JCL  
//* PRIVKEY=,  
// STEPLIB=  
//*  
//*****  
/* FIRST STEP DELETES ANY EXISTING SECURITY FILES.  
//*****  
//*  
//DELETE EXEC PGM=IEFBR14  
//CERT DD DSN=&HIGHQUAL..CERT.FIL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//PRIVATE DD DSN=&HIGHQUAL..PRIVATE.FIL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//SYMKEY DD DSN=&HIGHQUAL..SYMKEY.FIL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//ALIAS DD DSN=&HIGHQUAL..ALIAS.TBL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//CPLOOKUP DD DSN=&HIGHQUAL..CPLOOKUP.TBL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//PARTIC DD DSN=&HIGHQUAL..PARTIC.TBL,DISP=(MOD,DELETE,DELETE),  
// SPACE=(TRK,(1))  
//*  
//*****  
/* SECOND STEP DYNAMICALLY ALLOCATES AND DECOMpresses THE  
/* RUNTIME FILES FROM TDManager. THE 'RTMFILE', CREATED  
/* BY TDManager, MUST HAVE BEEN RECEIVED USING A BINARY FILE  
/* TRANSFER MECHANISM. THE 'APPCODE' IS REQUIRED IF THE RTMFILE  
/* IS ENCRYPTED WITH THE PARTICIPANT'S TDManager APPROVAL CODE.  
/*  
/* THE SECURITY FILES CREATED BY THIS STEP ARE:  
/* CERT.FIL  
/* PRIVATE.FIL  
/* SYMKEY.FIL  
/* ALIAS.TBL  
/* CPLOOKUP.TBL  
//*****  
//*  
//DECOMP EXEC PGM=DECOMP,REGION=2M,  
// PARM='NOUNCOMP NOVERIFY KEEPSIGS KEY=&APPCODE'  
//STEPLIB DD DSN=&STEPLIB,DISP=SHR
```

```
//DCMPLOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DATAIN DD DSN=&RTMFILE,DISP=SHR
///*
//*****THIRD STEP PROCESSES THE PRIVATE KEY SECURITY FILE
//* AND DYNAMICALLY ALLOCATES/WRITES THE PASSPHRASE FILES
//*
//* IF THIS STEP FAILS, THEN RERUN THE ENTIRE JOB.
//*****
//*
//IMPPRIV EXEC PGM=IMPPRIV,REGION=2M
//STEPLIB DD DSN=&STEPLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//PRIVKEYS DD DSN=&HIGHQUAL..PRIVATE.FIL,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,DISP=(NEW,DELETE),
//        UNIT=SYSDA,SPACE=(TRK,(1,1))
//*
//* THE FOLLOWING DATASET IS ONLY REQUIRED IF THE RSA KEYS
//* WERE GENERATED IN BATCH VIA THE 'GENKEYS' PROGRAM
//* OTHERWISE, COMMENT OUT THE DD STATEMENT
//*
//*PRIVKEY DD DSN=&PRIVKEY,DISP=SHR
//        PEND
//*
//*****EXECUTE THE PROCEDURE TO INSTALL THE SECURITY RUNTIME FILES.
//*
//* PROVIDE THE FOLLOWING VALUES ON THE JCL BELOW TO RUN THE PROC:
//*
//*   1. THE NAME OF THE COMPRESSED RUNTIME FILE FROM THE
//*      TDMANAGER.
//*   2. THE HIGH-LEVEL QUALIFIER TO USE FOR NAMING THE DECOMPRESSED
//*      SECURITY FILES. YOU MUST PROVIDE THE HIGH-LEVEL QUALIFIER
//*      IN TWO PLACES -- ON THE 'HIGHQUAL' PARAMETER OF THE 'PROC'
//*      STATEMENT AND ON THE 'DSN=' KEYWORD IN THE 'DECOMP.SYSIN'
//*      STATEMENTS.
//*   3. YOUR APPROVAL CODE PROVIDED BY THE HUB TRADING PARTNER.
//*   4. THE NAME OF THE 'PRIVKEY' FILE CREATED BY THE 'GENKEYS'
//*      UTILITY. IF YOU DID NOT USE 'GENKEYS' TO GENERATE YOUR
//*      RSA KEY PAIR, THEN COMMENT OUT THE 'PRIVKEY' PARAMETER
//*      ON THE PROC STATEMENT AS WELL AS THE 'PRIVKEY' DD STATEMENT.
//*   5. THE NAME OF THE STEPLIB WHERE THE 'DECOMP' AND 'IMPPRIV'
//*      PROGRAMS ARE INSTALLED.
//*****
```

```
/*
//STEP010 EXEC IMPORT,
//           RTMFILE=rtmfile,
//           HIGHQUAL=highqual,
//           APPCODE=0123456789ABCDEF,
//           PRIVKEY=privkey,
//           STEPLIB=TDCLIENT.LOADLIB
//*
//DECOMP.SYSIN    DD *
DSN=highqual      # <== HIGH-LEVEL QUALIFIER(S) FOR SECURITY FILES
LRECL=84          # <== RECORD LENGTH -- DO NOT CHANGE!
BLKSIZE=4204      # <== BLOCK SIZE
RECFM=VB          # <== RECORD FORMAT -- DO NOT CHANGE!
TRACKS            # <== ALLOCATION UNITS
PRIMARY=1         # <== PRIMARY ALLOCATION
SECONDARY=1        # <== SECONDARY ALLOCATION
```

Installation and Key Exchange is complete.

7.2 Utilizing TDClient to Execute Send/Receive Transfers

In order to run a transfer on OS390 it is necessary to run the BATCH job found in the TDSAMP sample JCL library. Edit the Batch job accordingly to execute either the send or receive transfers. The batch job can call the TDClient parameters in three ways; CMDFILE PDS, CMDFILE instream, or Command-line JCL.

CMDFILES consist of a set of parameters that define the transfer. The CMDFILE parameters can be placed in a dataset or PDS member and called from the command-line parm field or it can be placed instream to the JCL.

7.2.1 Sample BATCH job executing a send transfer pointing to a command parameter PDS member:

Sample of a Send Transfer:

```
//TDCLIENT JOB , 'EASYACCESS', CLASS=A, MSGCLASS=X, NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT, REGION=4M,
//   PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB   DD DSN=TDLOAD.LOADLIB, DISP=SHR
//CMDFILE   DD DSN=CMDFILE.PDS(CMDSEND), DISP=SHR
//QDATA     DD DSN=DATA.TO.SEND, DISP=SHR
//*
```

```
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER      DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04      DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//ALIAS       DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
//SECFILE     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS      DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//*
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
//*
//CPFTPLOG  DD SYSOUT=*
//RESPLOG    DD SYSOUT=*
//COMPLOG    DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG      DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
//*
/*
```

Associated Command File:
Executing a Send Transfer

```
*****  
NETWORK="FEDEXNET X12.58"  
FTPPASSWD=PASSWORD  
TRANSFER=(NAME=SENDREMIT  
SEND=DD:QDATA  
SENDCLASS=REMIT  
SENDUSERID=GOFEDEXASYNC  
COMPRESS=Y  
CRLF=Y  
SENDASCII=Y)  
*****
```

The SENDCLASS value is determined by the type of data being sent. The CLASS information has been pre-defined in the easyacc.ini file referenced by the EASYACC DD. If you do not know what class you are sending, please contact Customer Support for assistance.

7.2.2 Sample BATCH job executing a send transfer using the Instream CMDFILE parameter

Execution of a Secured Send Transfer:

```
//TDCLIENT JOB , 'EASYACCESS', CLASS=A, MSGCLASS=X, NOTIFY=USERID  
/*  
//EASYACC EXEC PGM=TDCLIENT, REGION=4M,  
// PARM='CMDFILE=DD:CMDFILE'  
/*  
//STEPLIB DD DSN=TDLOAD.LOADLIB, DISP=SHR  
/*  
//CMDFILE DD *  
NETWORK="FEDEXNET X12.58"  
FTPUSERID=USERID  
FTPPASSWD=PASSWORD  
RESET  
TRANSFER=(NAME=SENDREMIT  
SENDUSERID=GOFEDEXASYNC  
SEND=DD:QDATA  
SENDCLASS=REMIT  
COMPRESS=Y  
SECURE=Y  
SENDASCII=Y  
FILTER=Y)  
/*  
//QDATA DD DSN=DATA.TO.SEND, DISP=SHR  
/*
```

```
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//ALIAS DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS DD DSN=CERT.FIL,DISP=SHR
//SECFILE DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//OUTMSG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//CPFTPLOG DD SYSOUT=*
//RESPLOG DD SYSOUT=*
//COMPLOG DD SYSOUT=*
//EAFTPLOG DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
//*
/*
```

7.2.3 Sample BATCH job executing a receive transfer using the CMDFILE

Execution of a Receive Transfer:

```
//TDCLIENT JOB , 'EASYACCESS' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE DD DSN=CMDFILE.PDS(CMDRECV),DISP=SHR
//RECFILE DD DSN=RECEIVE.FILE,DISP=(NEW,CATLG),
//                                     UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                                     LRECL=80,BLKSIZE=0,RECFM=FB
//*
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//ALIAS DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS DD DSN=CERT.FIL,DISP=SHR
//SECFILE DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//DATAXX DD DSN=REJECT.FILE,DISP=(NEW,CATLG),
//          UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=80,BLKSIZE=0,RECFM=FB
//OUTMSG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
```

```
//CPFTPLOG DD SYSOUT=*
//RESPLOG DD SYSOUT=*
//COMPLLOG DD SYSOUT=*
//EAFTPLOG DD SYSOUT=*
//EALOG DD SYSOUT=*
//EXFERLOG DD SYSOUT=*
//EAINILOG DD SYSOUT=*
//EAMEMLOG DD SYSOUT=*
///*
/* "LOG_FTP" IN EASYACC.INI */
/* "LOG_EASYACC" IN EASYACC.INI */
/* "LOG_XFER" IN EASYACC.INI */
/* "LOG_INI" IN EASYACC.INI */
/* "LOG_MEM" IN EASYACC.INI */
```

7.2.4 Receiving a file

RECEIVE NON-EDI TRANSFER:

```
*****
NETWORK="FEDEXNET X12.58"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=RECVDOMINV
RECEIVE=DD:RECVFILE
RECEIVECLASS=DOMINV
UNCOMP=Y
CRLF=Y)
*****
```

7.2.5 Example BATCH job executing a receive transfer using the Instream CMDFILE

```
//TDCLIENT JOB , 'EASYACCESS' , CLASS=A, MSGCLASS=X, NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
// PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE DD *
NETWORK="FEDEXNET X12.58"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=RECVDOMINV
RECEIVE=DD:RECVFILE
RECEIVECLASS=DOMINV
UNCOMP=Y
CRLF=Y)
//RECFILE DD DSN=RECEIVE.FILE,DISP=(NEW,CATLG),
//                                     UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                                     LRECL=80,BLKSIZE=0,RECFM=FB
//*
```

```
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER   DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//ALIAS     DD DSN=ALIAS.TBL,DISP=SHR
//CPLOOKUP DD DSN=CPLOOKUP.TBL,DISP=SHR
//PRIVKEYS  DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS  DD DSN=CERT.FIL,DISP=SHR
//SECFILE   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//           LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//           LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//           LRECL=256,BLKSIZE=0,RECFM=VB
//*
//DATAXX   DD DSN=REJECT.FILE,DISP=(NEW,CATLG),
//           UNIT=SYSDA,SPACE=(TRK,(5,5)),
//           LRECL=80,BLKSIZE=0,RECFM=FB
//OUTMSG   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//CPFTPLOG DD SYSOUT=*
//RESPLOG  DD SYSOUT=*
//COMPLOG  DD SYSOUT=*
//EAFTPLOG DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG    DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
//*
//*
```

8 Communication with IBM/IGN

8.1 Certificates

Certificates are provided by IBM. Please contact IBM for the user mailbox and certificate keys.

Parse the IBM supplied certificates using the CMDPARSE batch JCL located in the EASAMP PDS library created during install.

1. FTP the userid.001 or userid.PFX over to the mainframe in binary format. Use the userid.001 or userid.PFX file as input to the CMDPARSE batch job. Let FTP allocate the file attributes. It is imperative to transfer this file in binary mode when using FTP, otherwise, this file will be corrupted and unusable.

Example:

```
ftp> open os390
Connected to os390.
220-FTPD1 IBM FTP CS V2R8 at S390, 18:42:56 on 2001-02-14.
220 Connection will close if idle for more than 5 minutes.
User (p390mvs:(none)): uid
331 Send password please.
Password:
230 USERID is logged on. Working directory is "UID.".
ftp> bi
200 Representation type is Image
ftp> put userid.001 'data.set.pfx.key' rep
200 Port request OK.
125 Storing data set DATA.SET.PFX.KEY
250 Transfer completed successfully.
ftp: 5248 bytes sent in 0.02Seconds 328.00Kbytes/sec.
ftp>quit
```

8.2 CMDPARSE utility

(TDSAMP member (CMDPARSE))

The CMDPARSE utility will generate the KEY files necessary for encryption. The datasets generated from the CMDPARSE job will need to be referenced in the batch job JCL used when running stored transfers.

```
//CMDPARSE JOB , 'PGMRNAME' , CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
```

```
//*****
///* Sample JCL to run the CMDPARSE from the command line.
///*
///* This JCL allocates and processes the inbound certificates and
///* creates the required TDClient files.
//*****
//*
// SET HIGHQUAL=           High-level qualifier for new datasets
// SET EA2KMOVSC=          High-level qualifier for TDClient datasets
// SET PFXFILE=             Input file to CMDPARSE
// SET PASSWORD=            Password required to process input file
///*
//*****
///* The first step allocates the required output files.
//*****
//*
///*
//DELETE    EXEC PGM=IEFBR14
///*
//CERT      DD   DSN=&HIGHQUAL..CERT.FIL,DISP=(MOD,DELETE,DELETE),
//           SPACE=(TRK,(1,1))
///*
//PRIVATE   DD   DSN=&HIGHQUAL..PRIVATE.FIL,DISP=(MOD,DELETE,DELETE),
//           SPACE=(TRK,(1,1))
///*
///*
//*****
///* Execute CMDPARSE
///*
///* The password is passed as a parameter to this routine. It is
///* required.
//*****
//*
///*
//CMDPARSE   EXEC PGM=CMDPARSE,REGION=4M,
//           PARM='&PASSWORD'
///*
//STEPLIB    DD DISP=SHR,DSN=&EA2KMOVSC..LOADLIB
//SYSPRINT   DD SYSOUT=*
///*
//*****
///* Sample allocation for PFX data.
///*
///* Specify the name of your userid file on the PFXFILE DD statement.
//*****
```

```
/*
//PFXFILE DD DISP=SHR,DSN=&PFXFILE
/*
//***** Allocate the 'INI' files
/*
//* 'EASYACC' is a required file. It contains the EDINAME in
//*   the "SECURITY:" section of the INI file.
//*****
//EASYACC DD DISP=SHR,DSN=&EA2KMVSC..TDClient.INI
/*
//***** Define the files to be imported
/*
//* 'PRIVKEYS' is one of the TDCompress runtime files. It contains
//*   the TDClient users' private RSA key used to decrypt and
//*   digitally sign data.
//* 'PUBLKEYS' is one of the TDCompress runtime files. It contains
//*   the trading partners' public RSA certificates used to encrypt
//*   and verify digitally signed data.
//*****
//*
//PRIVKEYS DD DISP=(NEW,CATLG,DELETE),DSN=&HIGHQUAL..PRIVATE.FIL,
//           RECFM=FB,LRECL=80,BLKSIZE=6080,
//           SPACE=(TRK,(1,1))
//PUBLKEYS DD DISP=(NEW,CATLG,DELETE),DSN=&HIGHQUAL..CERT.FIL,
//           RECFM=FB,LRECL=80,BLKSIZE=6080,
//           SPACE=(TRK,(1,1))
//*
```

Installation and Key Exchange is complete.

8.3 Utilizing TDClient to Execute Send/Receive Transfers

In order to run a transfer on OS390 it is necessary to run the BATCH job found in the TDSAMP sample JCL library. Edit the Batch job accordingly to execute either the send or receive transfers. The batch job can call the TDClient parameters in three ways: CMDFILE PDS, CMDFILE instream, or Command-line JCL.

CMDFILES consist of a set of parameters that define the transfer. The CMDFILE parameters can be placed in a dataset or PDS member and called from the command-line parm field or it can be placed instream to the JCL.

8.3.1 Example BATCH job executing a send transfer pointing to a command parameter PDS member

Sample of a send transfer:

```
//TDCLIENT JOB , 'TDCLIENT' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE DD DSN=CMDFILE.PDS(CMDSEND),DISP=SHR
//QDATA DD DSN=DATA.TO.SEND,DISP=SHR
//*
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//PRIVKEYS DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS DD DSN=CERT.FIL,DISP=SHR
//SECFILE DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//OUTMSG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//CPFTPLOG DD SYSOUT=*
//RESPLOG DD SYSOUT=*
//COMPLOG DD SYSOUT=*
```

```
//EAFTPLOG DD SYSOUT=/*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG      DD SYSOUT=/*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=/*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=/*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=/*          /* "LOG_MEM" IN EASYACC.INI */
///*
//*
```

Sample Associated Command File For Executing a Send Transfer

```
*****
NETWORK="IGN-I/E SSL"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=SENDIGN
SEND=DD:QDATA
SENDCLASS=TEST
SENDUSERID=ACCT.USERID
COMPRESS=Y
CRLF=Y
SENDASCII=Y)
*****
```

The SENDCLASS is a value agreed upon by both trading partners.

8.3.2 Example BATCH job executing a send transfer using the Instream CMDFILE parameter

Sample of a Secured send transfer:

```
//TDCLIENT JOB , 'TDCLIENT', CLASS=A, MSGCLASS=X, NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT, REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB   DD DSN=TDLOAD.LOADLIB, DISP=SHR
//*
//CMDFILE   DD *
NETWORK="IGN-I/E SSL"
FTPUSERID=ACCT.USERID
FTPPASSWD=PASSWORD
RESET
TRANSFER=(NAME=IBMSEND
SENDUSERID=ACCT.USERID
SEND=DD:QDATA
SENDCLASS=TEST
COMPRESS=Y
SECURE=Y
```

```
SENDASCII=Y
FILTER=Y)
/*
//QDATA      DD DSN=DATA.TO.SEND,DISP=SHR
/*
//EASYACC    DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER      DD DSN=EXFER.INI.FILE,DISP=SHR
/*
//SYSUT1     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04     DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//                  LRECL=8192,BLKSIZE=0,RECFM=VB
/*
//PRIVKEYS   DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS   DD DSN=CERT.FIL,DISP=SHR
//SECFILE    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
/*
//EDIPDS     DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//                  LRECL=256,BLKSIZE=0,RECFM=VB
/*
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
/*
//CPFTPLOG   DD SYSOUT=*
//RESPLOG    DD SYSOUT=*
//COMPLOG    DD SYSOUT=*
//EAFTPLOG   DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG       DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG   DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG   DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG   DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
/*

```

8.3.3 Example BATCH job executing a receive transfer using the CMDFILE

Sample of a Receive transfer:

```
//TDCLIENT JOB , 'TDCLIENT' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
//*
//EASYACC EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
//*
//STEPLIB DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE DD DSN=CMDFILE.PDS(CMDRECV),DISP=SHR
//RECFILE DD DSN=RECEIVE.FILE,DISP=(NEW,CATLG),
//          UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=80,BLKSIZE=0,RECFM=FB
//*
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR
//EXFER DD DSN=EXFER.INI.FILE,DISP=SHR
//*
//SYSUT1 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//PRIVKEYS DD DSN=PRIVATE.FIL,DISP=SHR
//PUBLKEYS DD DSN=CERT.FIL,DISP=SHR
//SECFILE DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//EDIPDS DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//SEDILOG DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),
//          LRECL=256,BLKSIZE=0,RECFM=VB
//*
//DATAXX DD DSN=REJECT.FILE,DISP=(NEW,CATLG),
//          UNIT=SYSDA,SPACE=(TRK,(5,5)),
//          LRECL=80,BLKSIZE=0,RECFM=FB
//OUTMSG DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
///*
//CPFTPLOG DD SYSOUT=*
//RESPLOG DD SYSOUT=*
//COMPLOG DD SYSOUT=*
//EAFTPLOG DD SYSOUT=*          /* "LOG_FTP" IN EASYACC.INI */
//EALOG    DD SYSOUT=*          /* "LOG_EASYACC" IN EASYACC.INI */
//EXFERLOG DD SYSOUT=*          /* "LOG_XFER" IN EASYACC.INI */
//EAINILOG DD SYSOUT=*          /* "LOG_INI" IN EASYACC.INI */
//EAMEMLOG DD SYSOUT=*          /* "LOG_MEM" IN EASYACC.INI */
///*
//
```

8.3.4 Receiving a FILE:

RECEIVE NON-EDI TRANSFER:

```
*****
NETWORK="IGN-I/E SSL"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=IBMRCV
RECEIVE=DD:RECVFILE
RECEIVECLASS=TEST
UNCOMP=Y
CRLF=Y)
*****
```

8.3.5 Example BATCH job executing a receive transfer using the Instream CMDFILE

```
//TDCLIENT JOB , 'TDCLIENT' , CLASS=A,MSGCLASS=X,NOTIFY=USERID
///*
//EASYACC   EXEC PGM=TDCLIENT,REGION=4M,
//  PARM='CMDFILE=DD:CMDFILE'
///*
//STEPLIB   DD DSN=TDLOAD.LOADLIB,DISP=SHR
//CMDFILE   DD *
NETWORK="FEDEXNET X12.58"
FTPPASSWD=PASSWORD
TRANSFER=(NAME=RECVDOMINV
RECEIVE=DD:RECVFILE
RECEIVECLASS=DOMINV
UNCOMP=Y
CRLF=Y)
//RECFILE   DD DSN=RECEIVE.FILE,DISP=(NEW,CATLG),
```

```
//                                     UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//                                     LRECL=80,BLKSIZE=0,RECFM=FB  
/*  
//EASYACC DD DSN=EASYACC.INI.FILE,DISP=SHR  
//EXFER   DD DSN=EXFER.INI.FILE,DISP=SHR  
/*  
//SYSUT1   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
//WORK01   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
//WORK02   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
//WORK03   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
//WORK04   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
//EASTATUS DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=8192,BLKSIZE=0,RECFM=VB  
/*  
//PRIVKEYS DD DSN=PRIVATE.FIL,DISP=SHR  
//PUBLKEYS DD DSN=CERT.FIL,DISP=SHR  
//SECFILE   DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),  
//           LRECL=256,BLKSIZE=0,RECFM=VB  
/*  
//EDIPDS   DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(1,1,10)),  
//           LRECL=256,BLKSIZE=0,RECFM=VB  
//SEDILOG  DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(1,1)),  
//           LRECL=256,BLKSIZE=0,RECFM=VB  
/*  
//DATAXX   DD DSN=REJECT.FILE,DISP=(NEW,CATLG),  
//           UNIT=SYSDA,SPACE=(TRK,(5,5)),  
//           LRECL=80,BLKSIZE=0,RECFM=FB  
//OUTMSG   DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
/*  
//CPFTPLOG DD SYSOUT=*  
//RESPLOG  DD SYSOUT=*  
//COMPLLOG DD SYSOUT=*  
//EAFTPLOG DD SYSOUT=*      /* "LOG_FTP" IN EASYACC.INI */  
//EALOG    DD SYSOUT=*      /* "LOG_EASYACC" IN EASYACC.INI */  
//EXFERLOG DD SYSOUT=*      /* "LOG_XFER" IN EASYACC.INI */  
//EAINILOG DD SYSOUT=*      /* "LOG_INI" IN EASYACC.INI */  
//EAMEMLOG DD SYSOUT=*      /* "LOG_MEM" IN EASYACC.INI */  
/*
```