Securing z/OS FTP Transmissions

Michael W. Moss
Value-4IT Limited

6 November 2019
Session EK

Securing z/OS FTP Transmissions: Agenda

- *Communications Server FTP Introduction & Overview*
- *IBM z/OS FTP: Protocols Functional Comparison*
- *z/OS FTP Logs & Analysis Overview*
- *z/OS FTP, FTPS & SFTP Functions Overview*
- *z/OS FTP Method Comparison: Security vs. Compliance*
- *IBM z/OS FTP Security Deployment: General Overview*
- *IBM z/OS FTP: Data Sources For Monitoring FTP Activity*
- *Are FTPS & SFTP GDPR Compliant? Managed File Transfer (MFT)*
- *Data Evolution: Cost-Availability-Capacity-Performance-Security*
- *Securing z/OS FTP Transmissions: Conclusion*
- *FTP Demo (If Time Allows)*

IBM TCP/IP General Availability (GA) Reprise

June 1992 - IBM TCP/IP For MVS Version 2.2 provides the user of MVS (base OS) the capability to participate in a multivendor network using the TCP/IP protocol set:

- ***CICS-TCP/IP Socket Interface***: Allow CICS Client or Server applications to interoperate with applications on processors (platforms) attached to a TCP/IP network.
- ***LPR/LPD Printing Support***: Remote printing client server support. The line printer client (LPR) sends print data to a line printer daemon (LPD) on a specified server host printer.
- ***Telnet & FTP SMF Record Support***: The TELNET Server provides SMF records identifying the local user id, local & remote IP addresses, log-on/off date/time & VTAM LU used for log-on. The FTP server provides SMF records for the Append, Delete/Mdelete, Get/Mget, Put/Mput & Rename client services, plus failed login attempts.
- ***FTP Security Exits***: Check IP & PORT Addresses, Check USER & Password, Check Command & Check JES usage; based on a premise of accepting or rejecting the FTP request.
- ***Db2 FTP SQL Query***: Remote TCP/IP Db2 SQL query transmission; results retrieved via FTP.
- ***FTP Password Change***: Allow the user to change the RACF (ACF2, TSS) account password.

For ~30 years, the Mainframe has been an IP network node, with FTP functionality

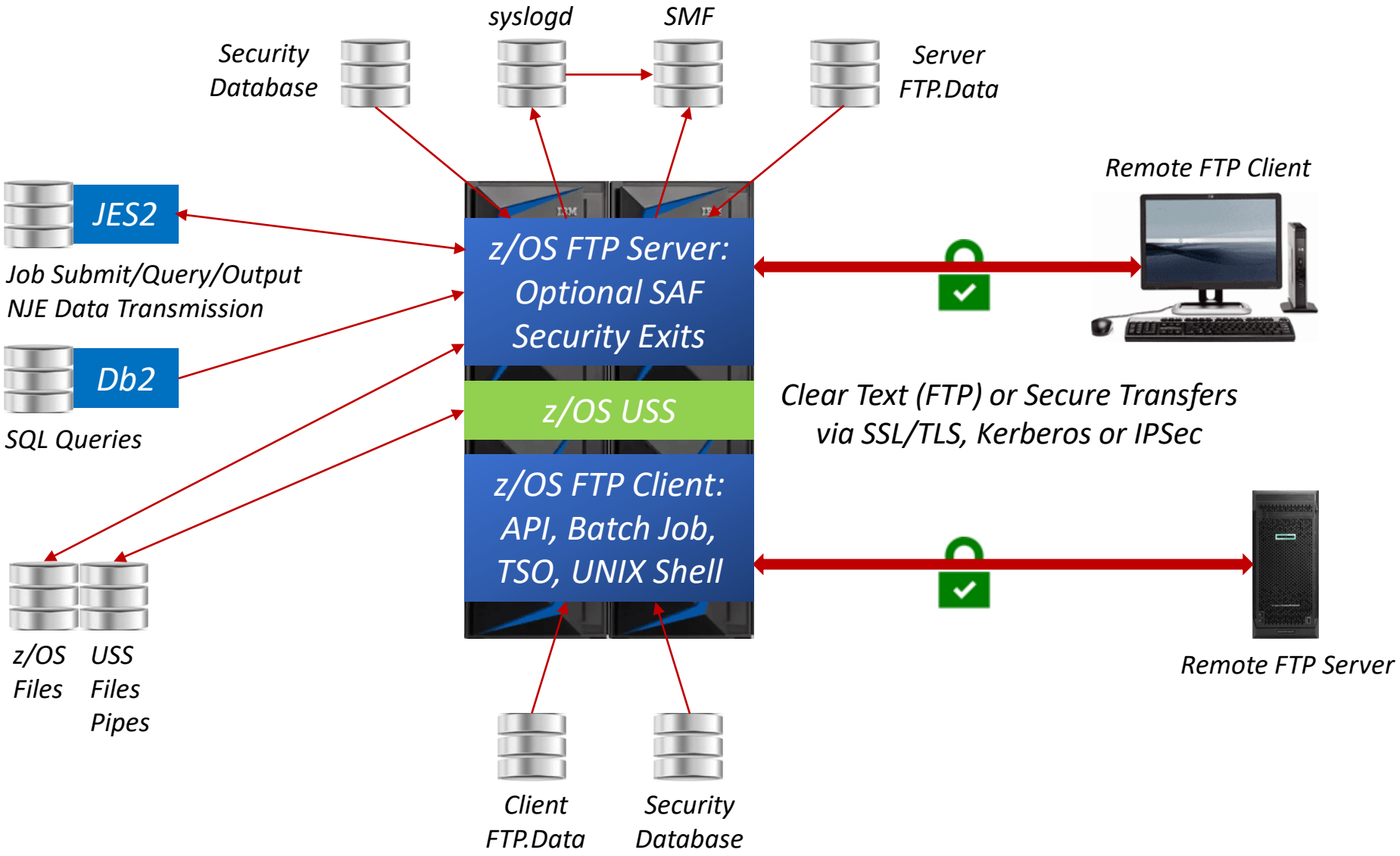
IBM z/OS Communications Server FTP Introduction

FTP is a z/OS UNIX System Services (USS) application. It can be started within an z/OS environment, but does not remain, immediately forking into the z/OS USS environment.

The z/OS FTP server includes a daemon & server process. The daemon process starts when FTP is started (E.g. START FTPD) & listens for connection requests on a specific port (I.E. 21 unless otherwise specified). When the daemon accepts an incoming connection, it creates a new process (server address space) for the FTP server, which handles the connection for the rest of the FTP login session; each FTP login session has its own server process. The server process inherits the accepted connection from the daemon process. This connection is called the control connection. The server receives commands from the client, sending replies to the client using the control connection. The control connection port is the same as the daemon's listening port.

The client & server use a different connection for transferring data; the data connection. By default, the data port ID is one less than the control connection port (E.g. Control 21, Data 20). An FTP client can override the default data port, directing the server to run in passive mode. In passive (PASV) mode, the server uses an ephemeral (transient) data port. Passive mode is requested by firewall friendly clients & by clients initiating three-way data transfers.

IBM z/OS Communications Server FTP Schematic



IBM z/OS FTP: Protocols Functional Comparison

FTP Protocol Type: Function Description	***FTP (No Security): RFC 959***	***FTPS (SSL/TLS): RFC 959/RFC 4217***	***FTP (IPSec): Any RFC Standard***	***SFTP (SSH) RFC 4253: IBM Ported Tools***
User ID & Password Secured	No	Yes	Yes	Yes
Data In-Flight Encryption	No	Yes	Yes	Yes
z/OS USS File Support	Yes	Yes	Yes	Yes
z/OS (MVS) File Support	Yes	Yes	Yes	No (*3rd Party Support*)
System Z Encryption Features Support (CPACF, ICSF)	No	Yes	Yes	Yes
Partner Authentication: Locally Stored Public Key Copies	No	No	Yes (Pre Shared Key)	Yes
Partner Authentication: X.509 Certificates (E.g. SSL/TLS)	No	Yes	Yes	No (*3rd Party Support*)
SAF Key Ring, ICSF Usage	No	Yes	Yes	Yes
FIPS 140-2 Support	No	Yes (z/OS 1.11)	Yes (z/OS 1.12)	No (*3rd Party Support*)
Mutual (2-Way) Authentication Support	No	Yes	Yes (@ IP Address Level)	No (*3rd Party Support*)

3rd party support delivered by specialized ISV software (E.g. Co:Z SFTP, VFTP-SSH)

NB. TFTP (Trivial File Transfer Protocol) excluded, due to EOS @ z/OS 2.2 & absence of client user identification & authentication.

How to choose, FTPS or SFTP? By default the IBM z/OS FTP server doesn't support SFTP…

z/OS FTP Logs & Analysis: Is SMF Data Good Enough?

In theory, by default FTP can collect SMF 118 & 119 records; for performance & usability, only use SMF 119 records, which collect more information with standard formatting:

- ***Subtype 3***: Client Transfer Completion (Includes cipher protocol, SSL/TLS, DES, FIPS 140, et al)
- ***Subtype 70***: Server Transfer Completion (Includes cipher specification, SSL/TLS, FIPS 140, et al)
- ***Subtype 71***: Daemon Configuration (Includes ciphersuite specification, SSL/TLS, DES)
- ***Subtype 72***: Server Logon Failure (Includes cipher protocol, SSL/TLS, DES, FIPS 140, et al)

FTP SMF logging has several blind spots, which could be exploited by savvy users, insiders or hackers:

- ***No Data Transfer Initiation***: No SMF record created, for trivial reasons such as command syntax, file not found are arbitrary. However, a SAF access failure for a sensitive file is a different matter…
- ***Batch FTP Transfer Initiation***: No SMF record created if the data is transferred successfully, but the job is cancelled before transfer success notification, a bad actor just needs to act accordingly…

Mostly, SMF records successful completed transfers. SMF can't be solely replied upon for suspicious client activity, sniffing the FTP server for secrets, requesting unknown files & directories until…

SMF is a good start, but let's think about what can go wrong, especially for sensitive data

z/OS FTP Logs & Analysis: Supplementing SMF Records

IBM introduced a “Check Confidence” feature in z/OS 1.7 allowing users to determine with some certainty that stream mode file structure transfer completed successfully, with 5 granularity levels:

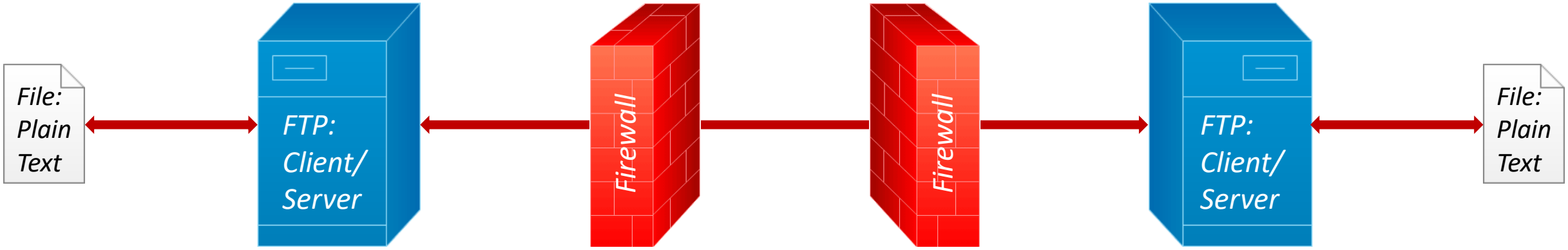
- ***High***: No error was detected.
- ***NoEOF***: An EOF marker was not found in STRUCT R or MODE B or C transfer.
- ***Low***: The client did not respond following the transfer or another error was reported.
- ***Unknown***: Only for outbound transfers & only set if checking is active.
- ***Inactive***: Only reported by the FTPOSTPR user exit.

There are three methods for conveying the Check Confidence levels:

- ***FTP Server Logging***: Via message EZYFS86I (FTP.DATA parameter FTPLOGGING set to TRUE).
- ***FTPOSTPR/EZAFCREP***: User exit (log replies sent out by FTP Server).
- ***FTP Client Messaging***: Via message EZA2108I.

FTCMDCHK (EZAFCCMD z/OS 2.2+) is an exit that can connect the server to SAF (ACF2, RACF, TSS) & can also be used to log incoming commands. In conjunction with FTPOSTPR (EZAFCREP z/OS 2.2+), this supplies a complete “Command & Reply” log, to highlight suspicious client activity.

z/OS FTP Function Overview: Introduction



Advantages:

- ✓ Ubiquitous (standard IP protocol)
- ✓ Easy-To-Use (common knowledge)
- ✓ Included In Base z/OS (zero cost)

Typical Usage:

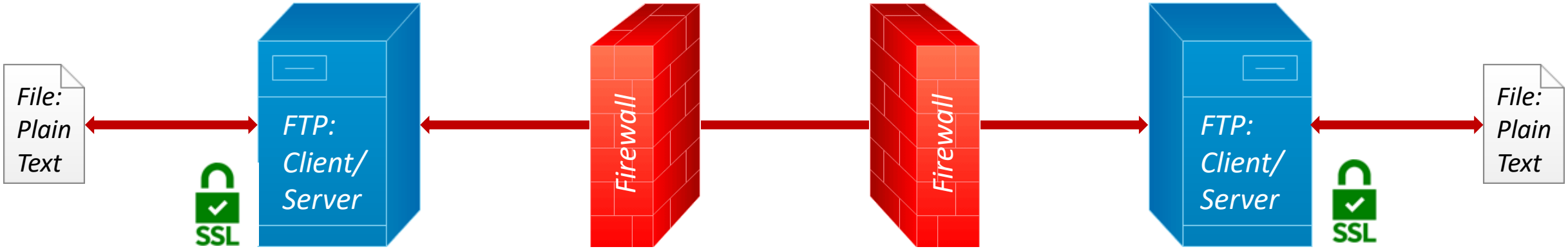
- ➢ Public Information
- ➢ Intranet (beware the insider attack)
- ➢ Anything & Everything (dangerous)

Disadvantages:

- ✖ Low Security (authentication)
- ✖ Firewall Unfriendly (too many ports)
- ✖ Data Xfer (compression, integrity)

Even ~30 years ago, a security conscious IBM Mainframe site shouldn't use basic FTP!

z/OS FTPS Function Overview: Introduction



SSL Encrypted Data & Commands via Implicit or Explicit Security

Advantages:

- ✓ RACF Key Ring Support
- ✓ X.509 Certificate, Kerberos Support
- ✓ Included In Base z/OS (zero cost)

Typical Usage:

- ➢ LPAR to LPAR (CPC/SYSPLEX)
- ➢ In-House System Z to System I
- ➢ Any Other FTPS Platform (in theory)

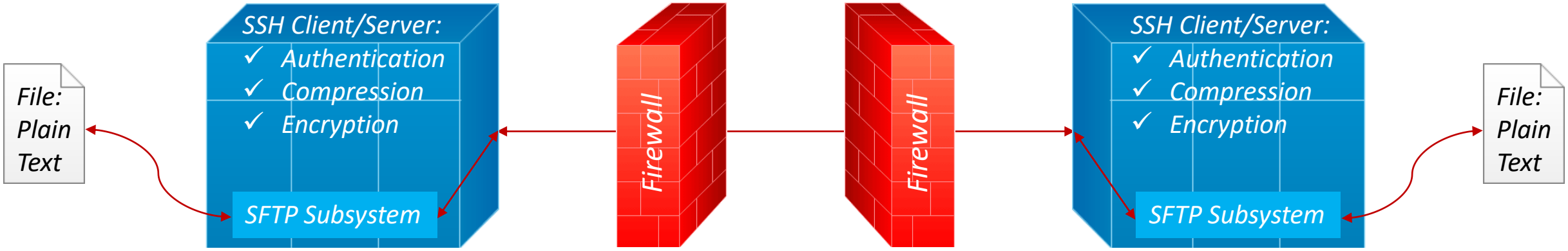
Disadvantages:

- ✖ Is FTPS Active @ Destination?
- ✖ Firewall Unfriendly (passive FTP)
- ✖ Does Destination Use Latest Cipher?

For implicit security, the entire connection is SSL/TLS protected. This method is outside normal FTP operations, as the client initiates SSL/TLS connection & the server needs to be made aware & participate. For explicit security, encryption is turned on by special command after the initial plaintext FTP connection set up. This command requirement adds a separate step for user/client action completion, changing the FTP protocol flow & user workflow.

Coming close to a usable & secure solution, but potentially limited to in-house usage…

z/OS SFTP Function Overview: Introduction



SSH Encrypted Command & Data Transfer Tunnel: Binary Packets

Advantages:

- ✓ End-to-End Encryption
- ✓ Inbuilt Data Integrity & Compression
- ✓ Distributed Systems Support
- ✓ Password Sniffing & MITM Defence

Typical Usage:

- ➢ Distributed Systems Interchange
- ➢ *Default - Anything & Everything: Why Not Use SFTP For All FTP?*

Disadvantages:

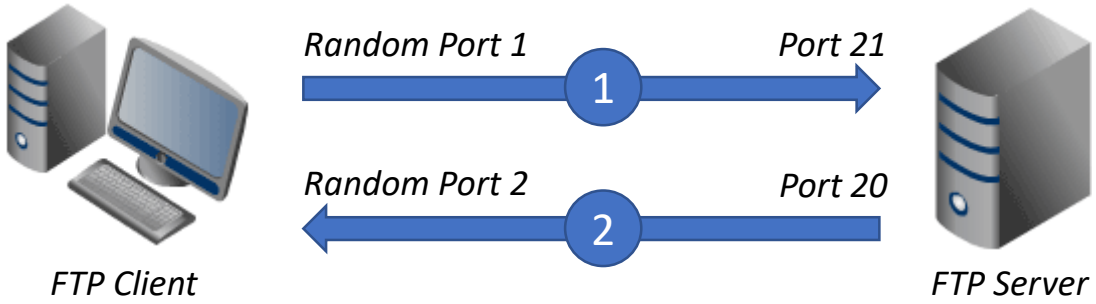
- ✖ Limited Use via z/OS Ported Tools
- ✖ Only Data-in-Flight Encryption
- ✖ Protocol Usability Learning Curve

A secure data-in-flight protocol, with potential 3rd party cost & usability considerations…

FTP Ports Consideration: Active vs. Passive Mode Review

GUIDE SHARE EUROPE UK REGION

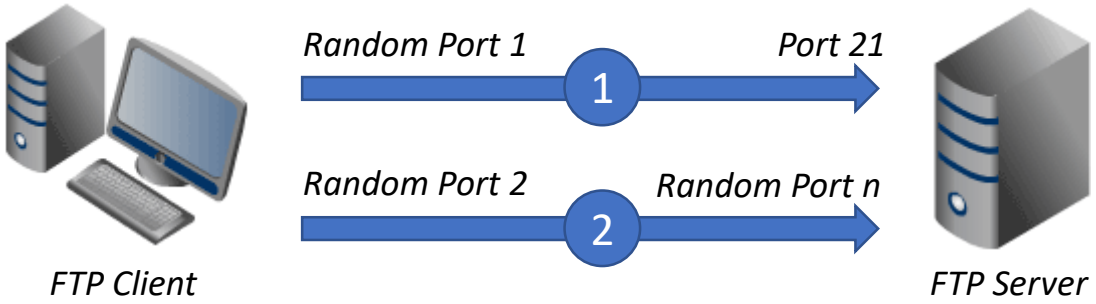
FTP Active Mode



1. File transfer client request via random port to server port 21, via PORT command, specifying client-side port ID for server connection. This is the data channel port, separate from the command channel port.

2. The server connects from port 20 to the designated data channel client port. Once connection is established, file transfers process via these client & server ports.

FTP Passive Mode



1. File transfer client request via random port to server port 21, via PASV command. The server replies indicating which random port number has been opened for data transfer.

2. The client connects from another random port number to the previously assigned server random port. Once connection is established, file transfers process via these client & server ports.

This simplified review of active vs. passive mode doesn't consider the pervasive use of a firewall. If passive mode is chosen, the system is more vulnerable to attacks, as FTP clients connect to multiple random server ports, requiring firewall access. The number of ports required for passive FTP is subjective, but FTP data transfers aren't always processed on a 1:1 basis. One FTP session might have n (E.g. 16) files, each requiring its own port. Some FTP clients support multiple data transfer connections, where n (E.g. 8) sessions are opened for faster transfer. Arguably active mode was designed when IBM Mainframes were more commonplace & cybersecurity attacks less so!

Which is simpler to manage; are there any other methods using a one secure port tunnel?

z/OS FTP Method Comparison: Security vs. Compliance

FTP Comparison	*FTP*	*FTPS*	*SFTP (SSH FTP)*	*Security Observations*
# Channels	2: 1 Command + 1 Data	2: 1 Command + 1 Data	1 (Command & Data)	1 secure channel for both data & authentication transfer is optimal.
Default Port #	21	990 Implicit SSL 21 Explicit SSL	22	FTP & FTPS require blocks of open ports for data channel creation (Firewall Issue).
Authentication	Userid & Password	Userid & Password And/Or Certificate	Userid & Password Or SSH Public Key	SSH authentication via a public-private key pair is considered the most secure.
Encryption	None	Command & Data SSL & TLS	SSH	FTPS security invalid for FTP servers not supporting SSL/TLS. SFTP is always secure.
Data Transfer	Plain Text	Implicit or Explicit: Plain Text (SSL/TLS Encryption)	Binary Data Packets	Plain text is human readable; easily hacked. Data packets are secure & transfer faster.
Connection	Unsecure	Unsecure or Secure	Secure	In a data privacy & compliance world, data transfers must always be secure.

GDPR Compliance Requirements: ***Automation*** (Transfer Success), ***Control*** (Management Console), ***Logging*** (Cradle-to-Grave Reporting), ***Data Security*** (Lifecycle Management, Secure Logs), ***Authentication*** (RBAC, MFA), ***Cryptography*** (In Flight & At Rest Encryption, Key Management), ***Secure Architecture*** (SAF Interface, Zero Unencrypted Data), ***Failover*** (Secure Business Continuity)

IBM z/OS FTP: Securing The FTP Server To-Do-List

1) Setting Up Security Framework

SERVAUTH Class, SAF Class APPL, TCPIP Profile PORT Keyword

2) Defining User Access

z/OS UNIX UID Allocation, FTP Server Login Access, TLS Authentication, FTP Exits

3) Port Of Entry Access

IP V4 via Terminal or SERVAUTH Class, IP V6 via SERVAUTH Class

4) Controlling z/OS UNIX File System Access

SERVAUTH Class

5) Preventing FTP Server Misuse

TCPIP Profile PORTCOMAND* & PASSIVE* Keywords

6) Controlling Db2 & JES Subsystem Usage

Bind Db2 Plan For Package EZAFTPMQ, FTCHKJES, JESINTERFACELEVEL & SDSF Considerations

7) Setting UP Reporting Information

FTP Server Logging, SMF & FTP Exits (FTCHKCMD, FTPOSTPR)

IBM z/OS FTP Security: Setting Up Security Framework

Activate the SERVAUTH Class in RACF:

SETROPTS CLASSACT(SERVAUTH) SETROPTS RACLIST(SERVAUTH)
SETROPTS GENERIC(SERVAUTH) SETROPTS RACLIST(SERVAUTH) REFRESH

Permit the FTPD User Read Access to the TCP/IP Stack (user) Access Control Profile:

RDEF SERVAUTH EZB.STACKACCESS.mvsname.tcpname.** UACC(NONE) warn (warning mode optional)
PE EZB.STACKACCESS.mvsname.tcpname.** CL(SERVAUTH) ID(FTPD) ACC(READ)

Note: If the SAF class APPL is activated & OMVSAPPL resource profile defined, the FTP deamon (user) requires read access, as does any user that performs a logon process to any FTP server originated ASID. This is a legacy SAF resource from OpenEdition MVS (4.3) the forerunner to z/OS UNIX Systems Services (USS).

Reserve the FTP daemon listening port the FTPD job by a PORT statement in the TCPIP PROFILE:

PORT 21 jobname (FTPD) SAF

Restrict Access to Port 21 for the FTP Daemon User Only:

RDEF SERVAUTH EZB.PORTACCESS.mvsname.tcpname.portname.** UACC(NONE)
PE EZB.PORTACCESS.mvsname.tcpname.portname.** CL(SERVAUTH) ID(FTPD) ACC(READ)

IBM z/OS FTP Security: Defining User Access

Define each user requiring FTP Server Login a z/OS UNIX UID:

It is not practical to assign a unique UID or GID for many users defined without OMVS segments who need access to z/OS USS (E.g. FTP). As of z/OS 1.11, allocate a unique UID for each user & a unique GID for each group that needs access to z/OS USS resources. Manual SC31-8775-20 states "the user ID that is associated with the FTP server STARTED class must have UID 0". Don't get confused with the FTP server itself & users, even so TCPIP PROFILE setting TLSMECHANISM ATTLS (SSL/TLS) can eradicate FTP SUPERUSER requirements…

Utilize FTP SERVAUTH resource for TLS Level 3 Authentication to control which users can login to FTP:

RDEF SERVAUTH EZB.FTPD.*.*.PORT21 UACC(NONE) - *FTPD daemon & Port 21*
PE EZB.FTPD.*.*.PORT21 CL(SERVAUTH) ID(FTPUser) ACC(READ)

Verify User access for every FTP session, secured or not in the TCPIP PROFILE:

VERIFYUSER TRUE

Consider the FTCHKPWD or FTCHKCMD exit to control FTP user access:

The FTCHKPWD exit is called after the FTP server receives the user ID & password from the FTP client, & before the server uses the password for authentication. However, FTCHKCMD is called whenever the FTP server completed any FTP command, where the user ID variable, among many others are passed. Moreover FTCHKCMD shares a scratchpad buffer area (256 bytes) with FTPOSTPR, making these two exits great candidates for FTP logging purposes, FTCHKCMD for what commands were requested & FTPOSTPR for the FTP activity logging.

IBM z/OS FTP Security: Port Of Entry Access

Plan ahead for TCPIP V6:

For IPv4 port of entry access security can be achieved via the SAF TERMINAL or SERVAUTH resources. For IPv6 connection partners automatically establish SERVAUTH access. Therefore for IPv4, use SERVAUTH resources.

Consider the yes/no or in/out network configuration options:

There are no golden rules or even rules of thumb for defining any kind of policy, but Keep It Simple Stupid (KISS) should always apply. We could define 2 FTP configurations, perhaps based on the FTP daemon server itself (E.g. FTPD1 & FTPD2), 2 sets of port ranges or 2 network access security zones. We could then have one set of policies for inside a firewall & the other for outside the firewall; one set of policies for standard file access & the other for sensitive files, & so on…

Example - Define 2 Network Access Security Zones in the TCPIP Profile:

```
NETACCESS
 9.24.104.0/24       ZONE1
 9.24.104.119/32     ZONE2
ENDNETACCESS
```

From a SAF SERVAUTH viewpoint, the Network Access Security Zone is EZB.NETACCESS.sysname.tcpname.zonename, where in this instance, the choices are either ZONE1 or ZONE2. A user requires read access to this profile for zone access. Keeping it simple, we could just use the default FTP sever port of 21 to eradicate port considerations…

IBM z/OS FTP Security: z/OS UNIX File System & Diagnostics Access

Safeguard the FTP UNIX file system:

FTP uses resource profile EZB.FTP.sysname.ftpdaemonname.ACCESS.HFS in the SAF SERVAUTH class, controlling access to the z/OS UNIX file system. Without profile access control, all users can access the z/OS UNIX file system.

Choose the default file system for FTP in the TCPIP Profile:

STARTDIRECTORY HFS

The STARTDIRECTORY statement specifies which file system (I.E. HFS or MVS) is initially used when a new FTP user logs in. This is another potential on/off switch for FTP processing; we might choose 2 FTP daemon servers (E.g. FTPD1 & FTPD2), allocating HFS as the default for one & MVS for the other. Please note, for ANONYMOUSFILEACCESS when the ANONYMOUSLEVEL is 3 or greater, the file system settings must be the same (I.E. HFS, MVS or BOTH).

Permit the FTP User Read Access to the z/OS FTP UNIX File System:

RDEFINE SERVAUTH EZB.FTP.sysname.ftpdaemonname.ACCESS.HFS
PERMIT EZB.FTP.sysname.ftpdaemonname.ACCESS.HFS CL(SERVAUTH) ID(ftpuser)

Control FTP diagnostics to appropriate technical support resources:

Use FTP resource EZB.FTP.sysname.ftpdaemonname.SITE.DUMP/DEBUG to control large output DUMP/DEBUG commands:

RDEF SERVAUTH EZB.FTP.*.*.SITE.* UACC(NONE)
PE EZB.FTP.*.*.SITE.* CLASS(SERVAUTH) ID(NETMGMT SYSPROG) ACCESS(READ)

IBM z/OS FTP Security: Preventing FTP Server Misuse

Any FTP server can be used by a client for disruptive purposes. A client can use the server to send random data to other servers, or a client can request the server be a passive server in a 3-way transfer. Any FTP client in PROXY mode with the FTP server can establish a data connection to any server listening to a port. This could cause severe disruption, the client could send significant amounts of unexpected data. Any malicious FTP client can attack or disrupt the server in a normal server-client connection, making the FTP server send a large amount of data to another application server listening to a specific port. Because the client is not sending the disruptive data, it is difficult to identify the client causing the problem:

FTP Server Status	***FTP.DATA Server Statement***	***FTP Server Operational Impact Description***
All PORT or ERPT Command Rejection	PORTCOMMAND REJECT	Disabling the PORT or EPRT commands prevents the server from being used to send random data to other servers, but the server loses some ability to transfer data in PROXY mode. If a client sends a PORT or EPRT command to initiate a proxy transfer, the command is rejected & the proxy transfer fails. If your client is not firewall friendly & does not implement the data transfer default port number & IP address, it cannot transfer files to & from the server.
All PORT or ERPT Command Rejection specifying typical low ports (1-1023)	PORTCOMMANDPORT NOLOWPORTS	The server cannot be used to send random data to servers listening on these ports. However, a rogue client can use the server to send random data to servers listening on other ports (E.g. 1024-65535). The server still supports data transfer in PROXY mode.
All PORT or ERPT Command Rejection specifying an alternative, not the client IP address	PORTCOMMANDIPADDR NOREDIRECT	With this combination, a client can only request PROXY mode data transfer between the server & a server using its own IP address. Transfers between client & server are not affected.
All PORT or ERPT Command Rejection specifying an alternative, not the client IP address or typical low ports (1-1023)	PORTCOMMANDPORT NOLOWPORTSPORT COMMANDIPADDR NOREDIRECT	With this combination, a client can only request PROXY mode data transfer between the server & a server using its own IP address; the port numbers are not well-known (E.g. 1-1023). The client cannot use PROXY mode to send random data to a server using its own IP address, listening to a well-known port.

IBM z/OS FTP Security: Controlling Db2 Subsystem Usage

Ask the most fundamental qualifying question; Do I need to allow user submitted Db2 SQL queries via FTP:

There are already a myriad of pervasive & fully supported ways for users to invoke Db2 SQL queries using functions & tools specifically designed to work with Db2. These include Db2 Interactive/Db2I (TSO/E & ISPF), SQL Processor Using File Input/SPUFI (via Db2I), Query Management Facility/QMF (Online or Batch), QMF for Workstation (Eclipse based workstation application) & numerous 3rd party ISV products. With all of this available functionality, enabling Db2 SQL queries via FTP seems extraneous to requirements:

Install the FTP Db2 SQL query function:

Bind a plan to invoke the package EZAFTPMQ in collection EZAFTPMQ, granting PUBLIC execution privileges for that plan. Specify the name of the plan using the DB2PLAN keyword in the TCPIP PROFILE (I.E. FTP.DATA), or the default is EZAFTPMQ. This FTP Db2 SQL query facility performs only SELECT operations on the Db2 tables; it does not perform UPDATE, INSERT, or DELETE operations. A sample job is provided in the FTOEBIND member of the SEZAINST Communication Server installation file & should be used to enable the FTP server & client to Db2 perform SQL queries. For secondary SQL queries authorization modify the Db2 DSN3SATH sample exit. The exit returns the primary AUTHID for requests originating from the FTP server. A summary of TCPIP PROFILE (I.E. FTP.DATA) changes required to install the FTP Db2 SQL query function is provided:

Set DB2 statement to specify Db2 subsystem name (E.g. DB2 DB2A – Default of DB2)
Set DB2PLAN to specify the Db2 plan used by the FTP server (E.g. DB2PLAN EZAFTPMQ – No default setting)
Set the SPREAD statement to control SQL output in spreadsheet format (E.g. SPREAD TRUE – Default of FALSE)
Set SQLCOL to specify output data column headings (E.g. SQLCOL ANY – Default of NAMES, LABELS as an option)

IBM z/OS FTP Security: Controlling JES Subsystem Usage

Ask the most fundamental qualifying question; Should users only manage their spool activity?

By default, the TCPIP PROFILE (I.E. FTP.DATA) JESINTERFACELEVEL setting is 1, an FTP user is allowed to submit jobs to JES, retrieve held output matching their logged-in user ID plus one character, & delete held jobs matching their logged-in user ID plus one character. Put another way, they can manage their own JES spool resources & no others. Arguably System Administrators are best placed to control the generic spool resource, periodically & automatically cleaning up legacy output (I.E. $TA JES command).

Install the FTP to JES interface function:

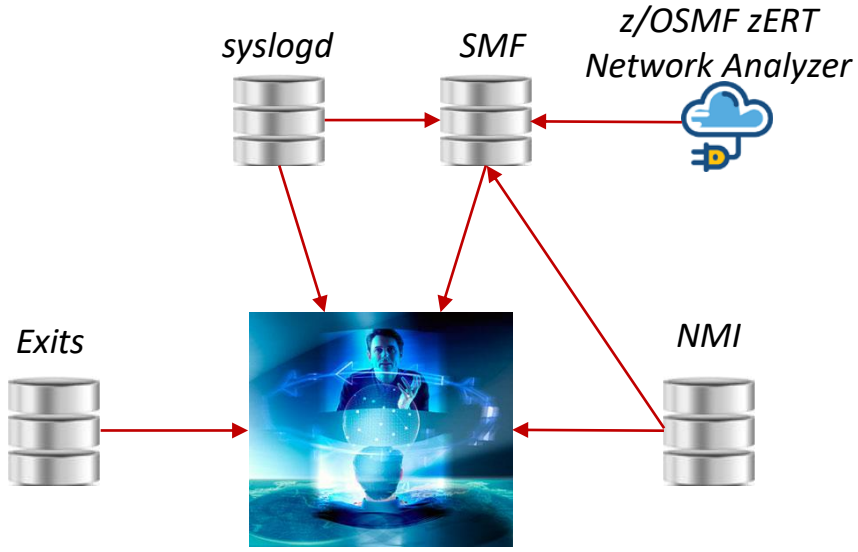
With JESINTERFACELEVEL 2, FTP users can retrieve & delete any system job for which they have SAF resource class JESSPOOL access & the ability to submit jobs is governed by the JESJOBS class SAF resource. JESINTERFACELEVEL 2 should only be specified if security measures ensure secure access to JES output. JESINTERFACELEVEL 2 uses the JES SAPI interface, requiring JESSPOOL READ authority to list job status or retrieve job output. The SAF controls used for JESINTERFACELEVEL 2 are essentially a subset of those used by SDSF, maintaining the security policies already in place for SDSF. The RACF Profile Name Format for the JESSPOOL class is localnodeid.userid.jobname.jobid.dsnumber.name. An FTP user can delete job output with ALTER access to the resource that matches the localnodeid.userid.jobname profile name (generic masking allowed); with READ access to this resource, the FTP user can list or retrieve the job output. The following TCPIP PROFILE (I.E. FTP.DATA) changes are required to enable the FTP to JES interface:

FILETYPE JES
JESINTERFACELEVEL 2

IBM z/OS FTP: Data Sources For Monitoring FTP Activity

syslogd outputs:

- Connectivity
- Authentication
- Access
- Allocation
- Deallocation
- Data transfer
- JES job submission
- SQL query
- Abnormal end
- Confidence (success level)



FTP SMF (119) outputs:

- Subtype 3: Client Transfer Completion
- Subtype 70: Server Transfer Completion
- Subtype 71: Daemon Configuration
- Subtype 72: Server Logon Failure

EZAFCCMD (FTCHKCMD)
EZAFCREP (FTPOSTPR)
FTP User Exits (Optional):

- EZAFCCMD: Pre ASCII conv. call for FTP commands Inspect, Modify; or end client FTP transfer.
- EZAFCREP: Post EBCDIC conv. Call for single or multiple line FTP replies, to inspect or end client FTP transfer.

Network Management Interface (NMI)
SYSTCPSM real-time FTP SMF outputs:

- Subtype 3, 70-72 (as per SMF 119)
- Subtype 100: Server Transfer Initiation
- Subtype 101: Client Transfer Initiation
- Subtype 102: Client Login Failure
- Subtype 103: Client Session Activity
- Subtype 104: Server Session Activity

A myriad of real-time & after the event data sources, but how do I manage this process?

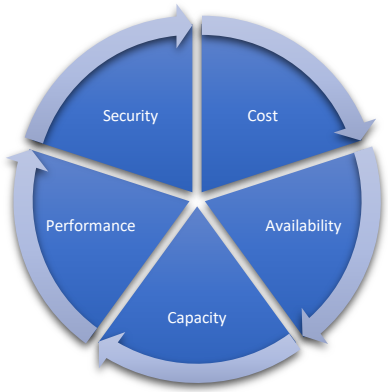
Are FTPS & SFTP GDPR Compliant? Managed File Transfer (MFT)

GUIDE SHARE EUROPE UK REGION

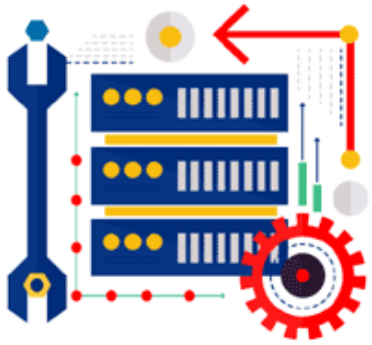
Ultimately we all must observe compliance mandates & quite simply, FTPS & SFTP are not GDPR compliant. If we perceive data protection as policy based & the enforcement & reporting thereof, ISO/IEC 27001 is a highly respected international security standard. If we consider ISO/IEC 27001 in terms of GDPR, we can evolve a Managed File Transfer (MFT) approach:

Security Mandate Requirement	***GDPR?***	***ISO 27001 #***	***File Transfer: Observations & Requirements***
Information Security Policies	Yes	A.5	Technology, tools or processes must safeguard file integrity checks, data deletion after receipt, & non-repudiation, where sender & receiver are both authorized & authenticated for data access. A tamper-evident (I.E. cannot be modified) audit trail is mandatory to record data integrity, delivery, authentication, non-repudiation & subsequent deletion of externally transmitted data files.
Access Control	Yes	A.9	User & administrator authentication is a pivotal cornerstone to robust security & compliance. File transfer systems must support a myriad of access control functions; centralized user directories, Role Based Access Control (RBAC), Single Sign-On (SSO) & Multi-Factor Authentication (MFA).
Cryptography	Yes	A.10	Data sharing systems must employ strong, state-of-the-art cryptographic mechanisms, enabling secure selection, distribution & encryption key protection. File transfer systems must ensure continuous data protection & integrity, in transit & at rest. Encryption keys have limited validity!
Secure Architecture	Yes	A.11	The underlying systems architecture must seamlessly integrate with existing security infrastructures & applications. These systems should safeguard there is no unencrypted data within the Demilitarized Zone (DMZ) or deliver DMZ termination of inbound requests for authentication & data transfer with a gateway proxy server.
Communications Security	Yes	A.13	Control & visibility of data flows & events are fundamental to effective security management & compliance validation. Systems must enable central visibility, control & prior authorization of all file transfers, retaining logs in tamper-evident repositories, safeguarding audit trail integrity.
Business Continuity	Yes	A.17	Secure business continuity safeguards the confidentiality, integrity availability of file transfers, at all stages throughout any failures, disasters or outages. Automatic, secure failover is essential to ensure that file transfers are either successful or continuously retried until complete.
Compliance	Yes	A.18	File transfer workflows must be automated to eradicate human error that could introduce data loss. File transfer solutions must support automatic forwarding, error correction, receipt confirmation, et al, for all data transfers.

Data Evolution: Cost-Availability-Capacity-Performance-Security



For 30+ years data importance has evolved, based on the available disk technology. In the late 1980's cost was the driver, as Mainframes used SLED not JBOD disks. In the early 1990's, when RAID JBOD implementations became commonplace, DR availability became a priority. Disk capacity then became a priority due to the data explosion of the mid 2000's. In the early 2010's, performance was the priority, processing ever increasing amounts of data, requiring pervasive flash & SSD resources. Finally, security is the focus, as the impact of encryption is nullified by previous disk developments. However, this evolution needed seamless transparent implementation solutions along the way…



Largely, Mainframe file allocations originate from batch JCL or software control statements. Changing these resources for a new storage function is onerous & time wasting, as another storage device or function will be delivered. To automate a transparent implementation of file allocation redirection, a simple to use method is required, redirecting & tweaking file allocations for business requirements, allocating the valuable business data asset to the most appropriate storage resource.

Initially DFSMS was such a disk reallocation mechanism, evolving over the decades, managing unified storage platforms in conjunction with self-tuning disk subsystems. The Virtual Tape Library (VTL) resource performs dynamic & automated file reallocation for tape data, to be staged or stored on disk, tape or cloud resources. FTP is arguably a legacy of DFSMS & VTL migration activities, eradicating tape for data exchange purposes. We have done it before for disk & tape, now FTP…

Securing z/OS FTP Transmissions: Conclusion

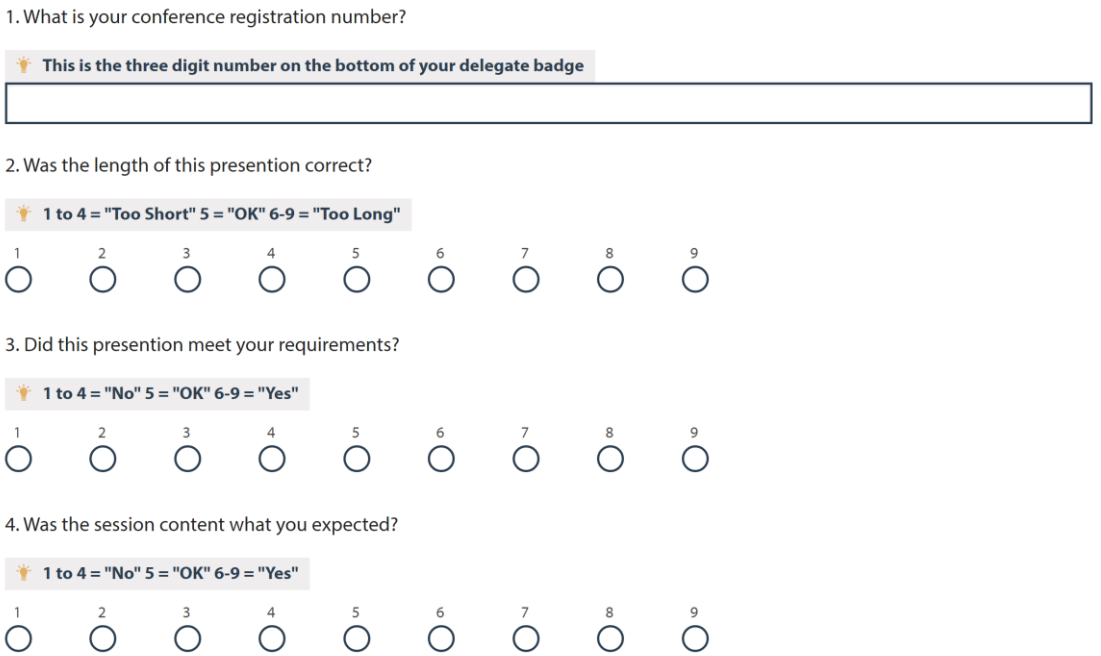
SAF Security: *When TCP/IP was first released FTP participating subsystems had internal security mechanisms such as DB2 (SYSIBM.SYSxxxAUTH Tables) & JES (SDSF ISFPARMS). TCP/IP also utilized the newly introduced UNIX Systems Services (USS) security mechanism & inevitably, ~25+ years ago, MVS people didn't always know a lot about UNIX & we might have made mistakes along the way. Now that SAF security is pervasive for all major z/OS functions & software, IBM or ISV, let's safeguard we always use SAF security resources accordingly.*

FTPS or SFTP: *For sure it's tempting to ask this question, but is it the right question? FTPS doesn't support all servers & doesn't define a standard for file name character sets (encodings); some might say that FTPS communication can be read & understood by a human, as being an advantage! SFTP has defined standards controlling all aspects of FTP operations, has only one shared connection (Control & Data) & the connection is always secured; SFTP communication is binary & cannot be understood by a human. However, IBM z/OS does not fully support SFTP, even with use of the IBM Ported Tools for z/OS: OpenSSH. Regardless, secure file transmissions are only one aspect of a fully-rounded and GDPR compliant Managed File Transfer (MFT) process. The FTP component of z/OS Communications Server might be zero cost, but this might be one of those instances, where a 3rd party ISV product is required to deliver a secure & compliant FTP environment.*

Being Secure via MFT: *Regardless of the solution, an organizations primary concern must be the security of their most valuable asset, namely data. By being the best they can be, an organization will become compliant & as always, a fully supported solution is best. To apply automation & the same robust security solution to each & every file transfer operation, a 3rd party ISV solution makes sense. Seamlessly prohibiting an insecure transfer, or converting it to a secure transaction, supplemented by a proactive Authentication, Authorization & Auditability process, appears to be a pragmatic way forward!*

Please submit your session feedback!

- Do it online at http://conferences.gse.org.uk/2019/feedback/ek
- This session is EK


1. What is your conference registration number?

This is the three digit number on the bottom of your delegate badge

2. Was the length of this presention correct?

1 to 4 = "Too Short" 5 = "OK" 6-9 = "Too Long"

1 2 3 4 5 6 7 8 9

3. Did this presention meet your requirements?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

4. Was the session content what you expected?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9