Securing z/OS® FTP Server with Control and Data Encryption

By Paul Newton

Today's IT personnel want to learn how to secure business data and intellectual property with easy-to-follow instructions. This how-to article explains the process of implementing z/OS® FTP server encryption using several TLS/SSL enable clients.

Introduction

The objective of this article is to explain how to implement z/OS® FTP server encryption using several TLS/SSL enable clients. An excellent source for TLS/SSL enabled clients is listed at the following Web site thanks to the work of Paul Ford-Hutchinson.


TLS, Transport Layer Security, is included with z/OS® V1R2 Communications Server and higher. TLS is a successor of SSL. Secure FTP session can be enabled by generating a certificate to be used by the FTP Server and placing several TLS-specific FTP statements in SYSFTPD data set. Entering gskkyman from the Unix System Service shell prompt is used to generate a certificate database, certificate and key.

Overview

A brief description of the secure FTP session establishment process will help with understanding of the certificate and TLS FTP parameters. A TLS secure FTP server will only begin to negotiate with a TLS-enabled client. The TLS-enabled FTP client sends an AUTH TLS or AUTH SSL to the FTP server. If the server does not receive AUTH TLS or AUTH SSL, negotiation stops. The FTP server responds to AUTH TLS or AUTH SSL with a certificate and public key. The client will compare the server certificate against client-imported certificates from trusted authorities. A match results in the client generating an encrypted session key from the public key provided by the server. If a certificate match is not found in the list of client-trusted certificates, then the client can

Figure 1: FTP Server Statements

EXTENSIONS AUTH_TLS
KEYRING /certificates/ftpServer.kdb
CIPHERSUITES SSL_3DES_SHA
CIPHERSUITES SSL_DES_SHA
CIPHERSUITES SSL_RC4_SHA
CIPHERSUITES SSL_RC4_MD5
SECURE_CNTLCONN PRIVATE
SECURE_DATACONN PRIVATE
SECURE_FTP REQUIRED
SECURE_LOGIN NO_CLIENT_AUTH
TLSTIMEOUT 600

Figure 2: BSDFTPD-SSL Client TLS Handshake

Security environment established - ready for negotiation
[TLSv1/SSLv3, cipher DES-CBC3-SHA, 168 bits]
Send password please.
Password:
IBMUSER is logged on.
Protection buffer size accepted
Data connection protection set to private
TLS/SSL protection of data connections on.
Remote system type is MVS.

Figure 3: z/OS® Client SYSFTPD Statements

EXTENSIONS AUTH_TLS
KEYRING /certificates/ftpClient.kdb
SECURE_DATACONN PRIVATE
FTP SERVER SETUP

Figure 1 shows the FTP server SYSFTPD statements that were used to complete the TLS handshake with various clients. The configuration 1) asks the client to use the highest available encryption level, 2) tells the client to encrypt data and control with the negotiated encryption level, 3) TLS/SSL authentication is required and 4) the server will not require client to send a trusted authority certificate. If the server configuration did require a certificate from the client, this certificate and key is typically issued by a trusted authority for import into both the server and client certificate database.

EXTENSIONS AUTH_TLS allows FTP server to support TLS authentication. When the server successfully processes the AUTH TLS or AUTH TLS-C command and completes the handshake with the FTP client, the control connection is protected by TLS. When the server successfully processes the AUTH TLS-P command and completes the handshake with the FTP client, the control connection is protected by TLS. The server also implicitly protects all data connections. When the server successfully processes the AUTH SSL command and completes the handshake with the FTP client, the control connection is protected by TLS. The server also implicitly protects all data connections.

KEYRING specifies the location of the certificate and key. This can be stored in Unix System Services certificate database created by gskkyman or SAF security package such as RACF.

Multiple CIPHERSUITE statements can be included in SYSFTPD data set. The client and server specify the list of encryption types that they support. The client and server negotiate which of the available ciphers is used for the data encryption by specifying the desired ciphers in order of preference. The actual cipher used is the best match between what the server supports and what the client requests. If the server does not support any of the ciphers that the client requests, the TLS handshake fails and the connection is closed.

SECURE_CNTLCONN sets control connection protection and SECURE_DATACONN sets data connection protection. Available settings are CLEAR, PRIVATE and SAFE. SECURE_DATACONN has an additional setting of NEVER. CLEAR means integrity protection or encryption is not required. PRIVATE means that integrity protection and encryption is required. SAFE means means integrity protection is required. NEVER means the data connection cannot be integrity protected or encrypted.

SECURE_FTP parameters, REQUIRED or ALLOWED, lets the client choose to authenticate or not authenticate.

SECURE_LOGIN parameters, NO_CLI-ENT_AUTH, REQUIRED or VERIFY_USER, specifies authentication level needed to complete TLS handshake. The server does not request a trusted certificate from the client when NO_CLIENT_AUTH is coded, however, REQUIRED means the client must send a trusted certificate which is known by the server to complete the TLS handshake. VERIFY_USER statement requires a SAF compliant product such as RACF®. VERIFY_USER requires a trusted certificate from the client and only permits authorized IDs to establish an FTP session.

The example server statement, SECURE_LOGIN NO_CLIENT_AUTH, permitted BSDftpd TLS/SSL capable client to complete the TLS handshake without importing any certificates. The commercially available TLS/SSL capable client did not import any certificates before TLS negotiation, however, a choice was given to accept or deny the server certificate as trusted during the TLS negotiation. The commercially available client needed to accept the server certificate as
trusted before TLS handshake would complete successfully. The z/OS® client required manual import of the certificate and key that was exported from the server certificate database to complete the TLS handshake successfully.

FTP CLIENT SETUP

Once the FTP server is configured to require secure ftp session, any FTP client not TLS/SSL capable that attempts a session will receive “Server requires authentication before user command.” The server is expecting client to send AUTH TLS or AUTH SSL to begin the conversation. An AUTH TLS or AUTH SSL sent from TLS/SSL capable client user ID will be accepted by the server. FIGURE 2 shows the messages from a successfully completed TLS handshake with BSDftpd-ssl client. BSDftpd-ssl client and the prerequisite OpenSSL package is available for download from http://bsdftpd-ssl.sc.ru/download.html.

z/OS® FTP client executed in batch, FIGURE 3, must include SYSFTPD DD reference to a data set containing statements found in FIGURE 4. TSO environment execution of FTP client, FIGURE 5, requires a preliminary allocate command associating the data set with file name SYSFTPD.

The z/OS® FTP client has a certificate database that can import z/OS® FTP server generated certificates. The certificate generated by the server was exported using gskkyman. The exported certificate was transferred to the client z/OS® where it was imported into the z/OS® client certificate database using gskkyman.

FIGURE 6 shows the messages from a successfully completed TLS handshake with a z/OS® client.

CERTIFICATE GENERATION USING Z/OS® FACILITIES

The z/OS® system needs an optional feature, Crypto Service SSL Base, installed. The article example references a Unix System Services file system mountpoint /certificates on the KEYRING statement. Entering gskkyman from the /certificates directory will produce a captive interface to guide you through the process. FIGURE 7 outlines the steps for creating the certificate database and generating certificate with key. Creation of certificate database sets permission allowing read only to owner, therefore, CHMOD 644 would permit read to TCP/IP address spaces.

SUMMARY

The reasons for securing data have increased as rapidly as the usage of the Internet. One consideration for implementing secure FTP transmissions is because the possibility exists those facilities could be used for transmitting sensitive data by someone who is unaware of the exposure. Hopefully this article provided some quick and easy steps to follow that could help with securing important business data and intellectual property.

NaSPA member Paul Newton is an IBM employee at the Dallas Technical Support Center.