COMPARISON OF FTP V/S FTPS

Subhasish Das ,VIT
Kusumakar Kashyap,VIT

Abstract: The File Transfer Protocol (FTP) is a standard network protocol used to transfer computer files from one host to another host over a TCP-based network, such as the Internet.FTP is built on a clientserver architecture and uses separate control and data connections between the client and the server.FTP users may authenticate themselves using clear sign-in protocol, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it. For secure transmission that protects the username and password, and encrypts the content, FTP is often secured with SSL/TLS (FTPS).

FTPS helps to encrypt and transfer private information within the constraints of regulatory requirements.

Many industries rely on the timely and effective transfer of files to provide services to consumers. For example, the healthcare industry requires exchanging sensitive information between healthcare providers, insurance providers, and eligibility services, to name a few. Regulatory requirements such as the Health Insurance Portability and Accountability Act (HIPAA) provide requirements for the use and disclosure of patients'private healthcare information (PHI). FTP services exchange information between caregivers and insurance companies, but the FTP protocol lacks the level of protection needed to meet regulatory requirements for the safeguarding of PHI. However, encrypting private information over the wire using FTPS helps meet this requirement.

Introduction

Working of FTP

FTP control connection created after TCP connection is established. Internal FTP commands are passed over this logical connection based on formatting rules established by the Telnet protocol.Each command sent by the client receives a reply from the server to indicate whether it succeeded or failed. A data connection is established for each individual data transfer to be performed. FTP supports either normal or passive data connections, allowing either the server or client to initiate the data connection. Multiple data types and file types are supported to allow flexibility for various types of transfers.FTP uses their liable Transmission Control Protocol (TCP) at the transport layer. An authentication system is used to ensure that only authorized clients are allowed to access a server. At the same time, feature sometimes called anonymous FTP allows an organization that wishes it to set up a general information server to provide files to anyone who might want to retrieve them.

The FTP server may support Active or Passive connections, or both. In an Active FTP connection, the client opens a port and listens and the server actively connects to it.

- Client opens up command channel from client port 2000(a) to server port 21(b).
- Client sends PORT 2001(a) to server and server acknowledges on command channel.
- Server opens up data channel from server port 20(b) to client port 2001(a).
- Client acknowledges on data channel.

In a Passive FTP connection, the server opens a port and listens (passively) and the client connects to it.

- Client opens up command channel from client port 2000(a) to server port 21(b).
- Client sends PASV to server on command channel.
- Server sends back (on command channel) PORT 1234(a) after starting to listen on that port.
- Client opens up data channel from client 2001(a) to server port 1234(a). • Server acknowledges on data channel.

Lesser-known Files Transfer Network Protocols:

The following list consists of the Lesser-known network FTP protocols:

SFTP (Simple File Transfer Protocol) is a network protocol of files transfer services, which based on TCP, developed by Ward Christensen (founder of Computerized Bulletin Board System) in 1977 for personal use. SFTP is an unsecured file transfer protocol .
AFP (Apple Filing Protocol) is a network protocol of files transfer services, which based on TCP, developed by Apple Inc in 1988 for connecting their Mac operating system.
Lynx is a network protocol of files transfer services which based on UDP, developed by Matthew Thomas in 1989 for batch data transmission.
9P (Plan 9 Filesystem Protocol) is a network protocol of files transfer services which based on TCP, developed by Bell Labs in 1990 for connecting the components of distributed operating system on UNIX, it was developed primarily for research purposes.
BiModem is a network protocol of files transfer services which based on UDP, developed by Erik Labs in 1995 for use in Bulletin Board Systems. (Also called BBS, it is an online service based on microcomputers running appropriate software).
SCP (Secure Copy Protocol) is a network protocol of files transfer services, which based on TCP, developed by ScriptLogic subsidiary of Quest Software Company in 2000 for commercial use. SCP is based on the Secure Shell (SSH) protocol to enforce encryption functionalities over any transmitted information .

Comparison Between Different Protocols And their Features

Protocol	Server Port	Transport
Bit Torrent	6881	TCP
Fast and secure protocol(FASP)	33001+	UDP
File Delivery over Unidirectional Trnsport(FLUTE)	4001	UDP
File Service Protocol(FSP)	Chosen by user	UDP
File Transfer Protocol(FTP)	20/>1023	TCP
Hypertext Transfer Protocol(HTTP)	80	TCP
Multicast File Transfer Protocol	5402	UDP
Simple Asynchronous File Transfer(SAFT)	487	TCP
SSH File Transfer Protocol	22	TCP

FTP - and its Limitations

File Transfer Protocol (FTP), provides medium of transfer of data between computers, i.e:server and a client on a network. While file size limitations are configurable with FTP, businesses typically use an FTP solution to eliminate the restrictions of physical media or mailbox quotas. However, FTP alone does not encrypt the transferred files. Establishing a connection with an FTP server leaves both authentication requests and the transferred data unencrypted over the wire. Any common packet-sniffing tool can read the exposed information. In industries with regulatory requirements such as PCI or HIPAA, protecting data and credentials to transfer data is not only necessary, but also required.

FTPS

When the FTP protocol was initially drafted security was not a concern. Since then many things have changed and sending data over any public network without encryption is considered very risky and in some cases prohibited. Regulations like PCI-DSS and HIPAA, for instance, contain provisions that require data transmissions to be protected by encryption.
In order to address this issue a set of security extensions to the original FTP protocol were proposed in RFC 2228 that protect FTP data as it travels over the network using SSL encryption.
This variant of FTP uses Transport Layer Security (SSL/TLS) to secure data. Take note though that SSL is very broken in terms of security and has now been disabled by most hosting providers (see our article on the Poodle Exploit for more information). There are two modes to the FTPS protocol, one is called “explicit” the other “implicit.”
In “explicit” mode, the client (you) can either initiate secured or un-secured transfers, providing the sever is configured to support this. In “Implicit” mode your FTP client must negotiate a secure transfer or fail the connection.

A standard FTP client communicates control with the server on port 21, FTPS normally uses port 990 and the data comes back on port 989, however FTPS can be implemented to use port 20 and 21 for both FTP and FTPS sessions. Generally Implicit mode is considered a deprecated method of negotiating TLS/SSL for FTP.
Because FTPS uses TLS as its underlying security it does require a valid TLS security certificate issued and signed by a Certificate Authority to be accepted without any error.

FTPS (FTP-SSL) is a real ftp that uses TSL/SSL to encrypt the control session and if required the data session. With FTPS the control session is always encrypted, but the data session might not be. The control session encrypted the authentication is protected. If you are NOT pre-encrypting the file, you want the data session encrypted so that the file is encrypted while the data is in flight. However, if you are pre-encrypting the file then you do not need to have the data connection encrypted as you do not need to add the overhead of encrypting the data connection, since the file is already encrypted. Understand that SFTP is SSH file transfer and FTPS is FTP with SSL, FTPS is a file transport layer on top of SSL or TLS. The FTPS adds SSL-enabled FTP send and receive capabilities, uses the FTP protocol to transfer files to and from SSL-enabled FTP servers.

Securing Data on the Wire with FTPS

Like its' HTTPS counterpart, FTPS includes the encryption necessary to protect the data across the wire. FTPS adds support for encryption to the original FTP protocol via SSL (Secure Sockets Layer) or TLS (Transport Layer Security). FTPS uses public key encryption and FTPS servers must provide an X.509 certificate signed by a trusted certificate authority. A plethora of FTPS solutions exist commercially to protect PHI and other sensitive data, including Ipswitch's MoveIT Transfer. A client/server FTPS implementation runs in one of two modes: implicit or explicit.

Implicit FTPS Mode

In implicit SSL mode a required SSL session is established between client and server before any data is exchanged. As it's name suggests, the use of SSL is implied and any connection attempt made by a client without using SSL are refused by the server. FTPS implicit SSL services generally run on port 990. Although still in use today, FTPS Implicit SSL is considered by many to be obsolete in favor of FTPS Explicit SSL.

Although considered deprecated, an FTP server in implicit mode requires a secure channel without giving the client the option to choose. No negotiation takes place with implicit connections, and a client is immediately expected to challenge the server with a clientHello or the connection is dropped. Implicit FTPS connections use port 990 for the control channel and 989 for the data channel.

Explicit FTPS Mode

In explicit SSL mode the client and server negotiate the level of protection used. This is very useful in that the server can support both unencrypted FTP and encrypted FTPS sessions on a single port. In an explicit SSL session the client first establishes an unencrypted connection to the FTP service. Prior to sending user credentials, the client requests that the server switch the command channel to an SSL encrypted channel by sending the AUTH TLS or AUTH SSL command. Upon successful setup of the SSL channel the client then sends user credentials to the FTP server. These credentials along with any other commands sent to server during the FTP session are automatically encrypted by the SSL channel. Similar to the way in which the command channel may be protected, the level of protection used on the data channel is negotiated between the client and server using the PROT command.

Explicit FTPS mode is the standard mode. It requires an FTP client to first explicitly request a secure connection and then to "step up" • to a mutually agreed upon encryption method. The control channel connection and data channel connection can step up separately. A secure control channel is established by using an AUTH SSL or AUTH TLS command, and this communication should be secured prior to authentication. After that, secure data channel can be established using the PROT command. To end the secure communication, the CDC (clear data channel) or CCC (clear command channel) commands can be used. As a result, the data can be encrypted when encryption is required, but it doesn't have to include the overhead of encryption when it isn't needed.

FTPS Or SFTP

While both include a combination of an asymmetric algorithm, a symmetric algorithm, and a key-exchange algorithm, FTPS uses x.509 certificates and most SFTP implementations use SSH keys. However, an SSH key does not verify the integrity of the key, nor the authenticity of the owner like a PKI-based solution does. FTPS is also not the same as FTP over SSH, tunneling the FTP traffic through a SSH connection. Pros and cons exist for both FTPS and SFTP, but diverse file-sharing solutions such as MoveIT Transfer support both so that you can determine which is right for your business needs.

Conclusion: The healthcare industry requires information exchange without compromising patients' PHI. While FTP alone doesn't provide the required protection - not only for patients' information safety, but also to meet the regulatory requirements - FTPS provides

the means to encrypt the data across the wire to secure the data transfer. Although FTPS solutions may not be free, a business will spend much less to secure the data than it will on a HIPAA violation, so an FTPS solution is well worth the cost.

REFERENCES

[1] A Novel Chaos-Based Voice Controlled FTP Tool-M.M. Ozturk,A.Akgul,Sezgin Kacar

[2] A Voice Controlled Web based FTP Tool- Muhammed MarufÖztürk,Ünal Çavuolu,Ahmet Zengin

[3] A File Transfer Protocol (FTP)-Michel Gien

[4] Overhead of FTPS and FTP over IPsec in IPv6 networks -Eng. N Pradeep Ruwan Nawarathne

[5] How To Properly Protect Data With FTPS- MISSY JANUSZKO

[6] M.P. Clark. Data Networks IP and the Internet.- 1st ed. West Sussex, England: John Wiley & Sons Ltd. 2003.

[7]Manual Pages: scp(1). Available at: http://www.openbsd.org/cgibin/man.cgi?query=scp&sektion=1 [Accessed March 30, 2010].

[8] T. Ylonen and C. Lonvick, Ed.The Secure Shell (SSH) Protocol Architecture. Internet Draft (RFC 4251), 2006.

[9] Christof Paar, Jan Pelzl, "**The Advanced Encryption Standard**", Chapter 4 of "Understanding Cryptography, A Textbook for Students and Practitioners". Springer, 2009.

[10] Triple DES Encryption Available at: http://www.tropsoft.com/strongenc/des3.html [Accessed February 5, 2010]