Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2022, 9(9):56-60

Research Article ISSN: 2394-658X

Enhancing Data Security with Secure FTP

Subhani Shaik

Programmer Analyst
Conch Technologies Inc, TN, USA
Email id – Subhani.shaik23@yahoo.com
469-531-6411

ABSTRACT

Secure FTP (File Transfer Protocol) stands as a cornerstone in modern data exchange, providing a robust mechanism for secure file transfers over networks. This abstract delves into the fundamental principles and key features of Secure FTP, shedding light on its pivotal role in safeguarding data integrity and confidentiality.

Secure FTP encompasses two primary implementations: FTPS (FTP Secure) and SFTP (SSH File Transfer Protocol), each offering distinct encryption methods and authentication mechanisms. FTPS extends the traditional FTP protocol with Transport Layer Security (TLS) or Secure Sockets Layer (SSL) encryption, while SFTP operates over Secure Shell (SSH), encrypting both commands and data.

Key features of Secure FTP include robust encryption to prevent unauthorized access and eavesdropping, flexible authentication options including passwords and public-key cryptography, and mechanisms to ensure data integrity during transmission. Moreover, Secure FTP solutions comply with industry standards and regulations, making them suitable for secure data exchange in various domains, including finance, healthcare, and e-commerce.

Organizations leverage Secure FTP for a myriad of purposes, including secure file transfers with external partners, automated workflows for data exchange, and compliance with regulatory requirements such as PCI DSS and HIPAA. With its portability, reliability, and adherence to security best practices, Secure FTP remains an indispensable tool in the modern data security arsenal, enabling organizations to exchange sensitive information with confidence and peace of mind.

Key words: Secure FTP, FTPS, SFTP, File Transfer Protocol, Encryption, Transport Layer Security (TLS), Secure Sockets Layer (SSL), Secure Shell (SSH), Authentication, Data Integrity, Data Security, Compliance, Data Exchange, Confidentiality, Authentication Mechanisms, Public-key Cryptography, Regulatory Requirements, PCI DSS, HIPAA, File Transfer.

INTRODUCTION TO SECURE FTP

Secure FTP (File Transfer Protocol) is a protocol used to transfer files securely over a network, ensuring the confidentiality and integrity of data during transmission. In today's interconnected digital landscape, where data security is paramount, Secure FTP emerges as a fundamental tool for safeguarding sensitive information exchanged between systems, servers, and users.

Secure FTP encompasses two primary implementations: FTPS (FTP Secure) and SFTP (SSH File Transfer Protocol), each offering distinct encryption methods and authentication mechanisms to ensure secure file transfers. FTPS, an extension of the traditional FTP protocol, integrates Transport Layer Security (TLS) or Secure Sockets Layer (SSL) encryption, thereby encrypting data during transmission. It operates over two communication channels: a control channel for sending commands and a data channel for transferring files. FTPS supports explicit FTPS (FTPES), where security is initiated by the client, and implicit FTPS, where security is assumed from the outset.

On the other hand, SFTP operates over the Secure Shell (SSH) protocol, providing a secure channel for both commands and data transmission. SFTP encrypts data using SSH encryption techniques, ensuring confidentiality and integrity throughout the transfer process. Additionally, SFTP offers features such as remote file management and file permission control, enhancing its versatility and utility for secure file transfers.

Key features of Secure FTP include robust encryption mechanisms, flexible authentication options (such as passwords, public-key cryptography, and client certificates), and mechanisms to ensure data integrity and compliance with industry regulations. Organizations across various sectors leverage Secure FTP for a myriad of purposes, including exchanging sensitive files with external partners, automating file transfer workflows, and ensuring compliance with regulatory standards such as PCI DSS and HIPAA.

In summary, Secure FTP serves as a cornerstone in modern data exchange, providing organizations with a reliable and secure means of transferring files while mitigating the risks associated with unauthorized access, data interception, and tampering. By adopting Secure FTP solutions, organizations can enhance data security, streamline business processes, and foster trust and confidence among stakeholders in today's digital ecosystem.

WHY WE NEED SECURE FTP

Secure FTP is crucial for several reasons, primarily centered around ensuring the security, confidentiality, and integrity of data during file transfers. Here's why organizations opt for Secure FTP:

[1]. **Data Security:** Secure FTP encrypts data during transmission, preventing unauthorized access and eavesdropping. This encryption ensures that sensitive information, such as financial records, customer data, and intellectual property, remains protected from interception by malicious actors.

[2]. **Compliance Requirements:** Many industries, including finance, healthcare, and e-commerce, are subject to regulatory requirements that mandate the secure exchange of sensitive data. Secure FTP solutions help organizations comply with regulations such as PCI DSS (Payment Card Industry Data Security Standard) and HIPAA (Health Insurance Portability and Accountability Act) by ensuring data security and confidentiality.

[3]. **Authentication Mechanisms:** Secure FTP protocols support various authentication mechanisms, such as passwords, public-key cryptography, and client certificates. These authentication mechanisms help verify the identities of users and ensure that only authorized individuals can access and transfer files.

[4]. **Data Integrity:** Secure FTP solutions include mechanisms to ensure the integrity of data during transmission. By detecting and preventing tampering or modification of files, Secure FTP helps organizations maintain the accuracy and reliability of their data.

[5]. **Risk Mitigation:** Traditional FTP protocols transmit data in plain text, making them vulnerable to interception and data breaches. Secure FTP mitigates these risks by encrypting data, thereby reducing the likelihood of data breaches and protecting sensitive information from unauthorized access.

[6]. **Trust and Confidence:** Using Secure FTP solutions instills trust and confidence among stakeholders, including customers, partners, and regulatory authorities. By demonstrating a commitment to data security and privacy, organizations can build and maintain strong relationships with stakeholders and enhance their reputation in the marketplace.

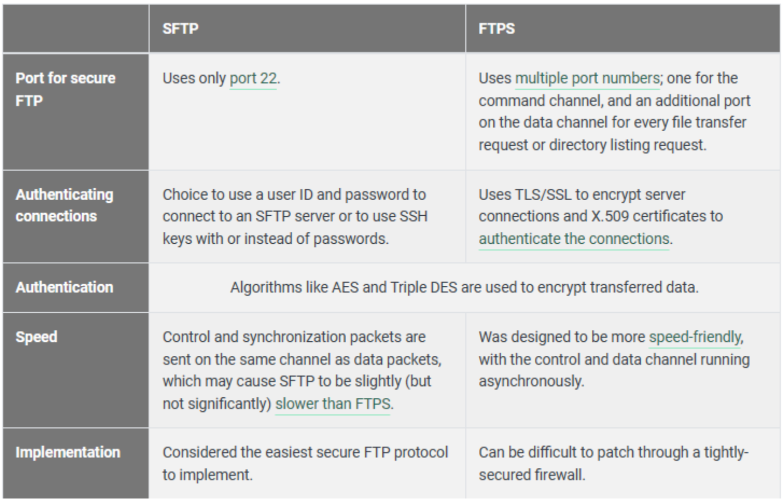
[7]. **Versatility and Flexibility:** Secure FTP solutions, such as FTPS and SFTP, are versatile and compatible with various operating systems and platforms. This versatility allows organizations to securely transfer files across different environments, including on-premises servers, cloud infrastructure, and remote locations.

[8]. **Operational Efficiency:** Despite its robust security features, Secure FTP solutions are designed to be user-friendly and easy to deploy. By streamlining file transfer workflows and automating routine tasks, Secure FTP helps organizations improve operational efficiency and productivity.

In summary, Secure FTP is essential for organizations seeking to protect sensitive data, comply with regulatory requirements, mitigate risks, and build trust with stakeholders. By implementing Secure FTP solutions, organizations can enhance data security, maintain compliance, and facilitate secure file transfers across their operations.

WHICH IS MORE SECURE: SFTP OR FTPS?

In summary, SFTP and FTPS are both secure FTP protocols with strong authentication options. Since SFTP is much easier to port through firewalls, however, I believe SFTP is the clear winner between the two.



	SFTP	FTPS
Port for secure FTP	Uses only port 22.	Uses multiple port numbers; one for the command channel, and an additional port on the data channel for every file transfer request or directory listing request.
Authenticating connections	Choice to use a user ID and password to connect to an SFTP server or to use SSH keys with or instead of passwords.	Uses TLS/SSL to encrypt server connections and X.509 certificates to authenticate the connections.
Authentication	Algorithms like AES and Triple DES are used to encrypt transferred data.	
Speed	Control and synchronization packets are sent on the same channel as data packets, which may cause SFTP to be slightly (but not significantly) slower than FTPS.	Was designed to be more speed-friendly, with the control and data channel running asynchronously.
Implementation	Considered the easiest secure FTP protocol to implement.	Can be difficult to patch through a tightly-secured firewall.

There are several different secure file transfer protocols that are, unfortunately, named in a very confusing way that often makes it difficult to distinguish one from another. The aim of this document is to provide some guidelines to make it easier to determine which is which.

COMMUNICATION PROTOCOLS

Basically, there are the following file transfer protocols around:

[1]. FTP - the plain old FTP protocol that has been around since 1970s. The acronym stands for "File Transfer Protocol". It usually runs over TCP port 21.

[2]. 2.SFTP - another, completely different file transfer protocol that has nothing to do with FTP. SFTP runs over an SSH session, usually on TCP port 22. It has been around since late 1990s. The acronym actually stands for "SSH File Transfer Protocol".

[3]. 3.SCP - a variant of BSD rcp utility that transfers files over SSH session. The SCP protocol has been mostly superseded by the more comprehensive SFTP protocol and some implementations of the "scp" utility actually use SFTP instead.

SECURE COMMUNICATION LAYERS

Additionally, there are the following two secure communication layers:

[1]. SSH - a protocol that allows establishing a secure channel between the local and the remote computer. Serves as an underlying channel for associated protocols such as secure shell, port forwarding, SFTP or SCP. While it is possible to run the (slightly modified) plain old FTP protocol over SSH, this is not very common, fortunately. File transfer over SSH is nearly always done using SFTP or SCP.

[2]. TLS - this is almost generally known primarily by its old name - SSL - and provides a way of securing otherwise unsecure protocols such as HTTP, SMTP, POP3 or FTP. Please note that SSL 3.1 is called TLS 1.0, and therefore TLS 1.0 is a newer version of the protocol than SSL 3.0, despite the lower version number. HTTP over SSL is often called HTTPS, and FTP over SSL is often called FTPS and has two variants, explicit (starts as an unencrypted FTP session and is secured on client request) and implicit (is secured right from the beginning and therefore needs a separate TCP port, usually 990). The implicit mode is deprecated, but still widely used.

SECURE FILE TRANSFER PROTOCOLS, OR FITTING IT ALL TOGETHER:

In an ideal world, the information above should be just enough. Unfortunately, this is not the case. The file transfer protocols are also referred to by other names, and even the names that only refer to a one single protocol are often mistakenly used for the wrong protocol by (understandably) confused authors.

[1]. FTP - should be only used for the plain old FTP protocol.

[2]. SFTP - should be only used for SFTP, the SSH file transfer protocol. However, people often shorten Secure FTP into SFTP - this is not correct, because the S in SFTP does not stand for Secure, but for SSH.

[3]. SFTP2 - this confusing name is used by some vendors to highlight the obvious fact that their SFTP protocol runs over SSH2. For all practical purposes, consider this to be a synonym of SFTP, because SSH1 has been deprecated for many years.

[4]. Secure FTP - this name is the most confusing, because it is used to refer to either of the two different protocols. Whenever this name is used, it is necessary to specify whether the SSH-based or SSL-based file transfer protocol is meant.

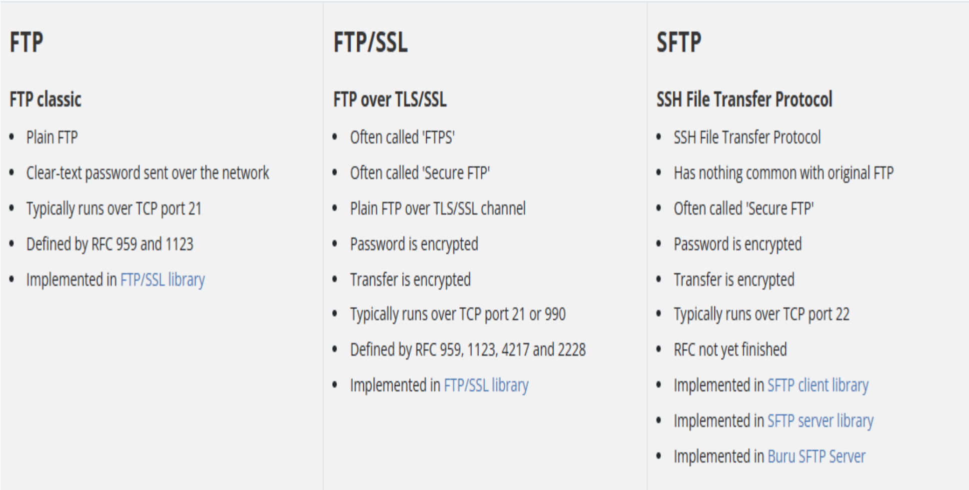
[5]. SSH FTP, FTP over SSH - fortunately, these names are not used very often. They usually refer to SFTP, the SSH file transfer protocol. Even though it is possible to run the (slightly modified) plain old FTP protocol over SSH, this is not very common.

[6]. FTP/SSL, FTP/TLS, FTP over SSL, FTP over TLS, FTPS - should be only used for FTP over TLS/SSL.

[7]. SFTP over SSL - although the SFTP protocol can utilize any underlying data stream, in practice SFTP over anything other that SSH is very rare. It is much more likely the term was used by mistake in place of either "SFTP over SSH" or "FTP over SSL".

[8]. SCP - should be only used for scp protocol/utility, a variant of BSD rcp. Some applications with SCP in its name now use SFTP by default instead - examples of this practice are WinSCP application and scp2 utility.

[9]. TFTP is yet another file transfer protocol different from any of the above.



FTP

FTP classic

- Plain FTP
- Clear-text password sent over the network
- Typically runs over TCP port 21
- Defined by RFC 959 and 1123
- Implemented in FTP/SSL library

FTP/SSL

FTP over TLS/SSL

- Often called 'FTPS'
- Often called 'Secure FTP'
- Plain FTP over TLS/SSL channel
- Password is encrypted
- Transfer is encrypted
- Typically runs over TCP port 21 or 990
- Defined by RFC 959, 1123, 4217 and 2228
- Implemented in FTP/SSL library

SFTP

SSH File Transfer Protocol

- SSH File Transfer Protocol
- Has nothing common with original FTP
- Often called 'Secure FTP'
- Password is encrypted
- Transfer is encrypted
- Typically runs over TCP port 22
- RFC not yet finished
- Implemented in SFTP client library
- Implemented in SFTP server library
- Implemented in Buru SFTP Server

WHO NEEDS SECURE FTP?

Secure FTP is ideal for organizations who need to send confidential files over the Internet or other unsecure networks. Here's a list of some areas where secure FTP can be useful.

[1]. Organizations operating in the healthcare industry and their business associates.

A. Guide to HIPAA Compliant File Transfers

B. Business Associates and HIPAA Compliant File Transfers

[2]. Organizations handling credit cards or debit cards.

A. Guide to PCI DSS Compliant File Transfers

B. Required MFT Server Password Settings for PCI DSS Compliance

[3]. Legal firms, paralegals, and their business associates

A. Ensuring Regulatory Compliance in eDiscovery File Transfers

B. Advantages of Using a Managed File Transfer Server During eDiscovery

C. How MFT Server File Sharing Minimizes Potential eDiscovery Costs

D. How to Share Files with a Virtual Paralegal

[4]. Manufacturers, suppliers, and CAD designers

A. A Faster Way to Send Big Files Essential to Manufacturing

[5]. Businesses who need to transfer large files to the cloud.

A. A Faster Way to Send Big Data to the Cloud

[6]. Businesses are starting to adopt a BYOD policy.

A. How Secure Mobile File Storage & Sharing Refines Your BYOD Strategy

REFERENCES

[1]. https://www.progress.com/resources/papers/sftp-server
[2]. https://www.rebex.net/kb/secure-ftp/
[3]. https://www.jscape.com/blog/secure-ftp-simplified
[4]. https://www.goanywhere.com/blog/sftp-vs-ftps-what-is-the-best-secure-ftp-protocol