

# FTPS

**FTPS** (commonly referred to as **FTP/SSL**) is a name used to encompass a number of ways in which [FTP](#) software can perform secure file transfers. Each way involves the use of a [SSL/TLS](#) layer below the standard FTP protocol to encrypt the control and/or data channels. It should not be confused with either [SSH file transfer protocol](#) (SFTP), or [FTP over SSH](#).

The most common uses of FTP and SSL are:

- *AUTH TLS, Explicit FTPS or FTPES*, named for the command issued to indicate that TLS security should be used. This is the preferred method according to [RFC 4217](#). The client connects using clear text on port 21 and may negotiate a secure TLS connection during the FTP setup or at any time thereafter. The server may allow non-encrypted FTP should negotiation fail. If the control channel is unencrypted, any subsequent data channels must also be unencrypted; if the control channel is encrypted, the subsequent data channels may be in clear or encrypted. (This is a requirement of the *AUTH* mechanism defined under [RFC 2228](#).) If the command channel is not encrypted, the protocol is said to be using a clear command channel (CCC). If the data channel is not encrypted, the protocol is said to be using a clear data channel (CDC). Encrypted data channels and encryption on the control channel can be set up and torn down by the client at any time.
- *Implicit FTPS* is an older, but still widely implemented, style in which the client connects to a different control port, and an SSL handshake is performed before any FTP commands are sent. The entire FTPS session is encrypted. Implicit FTPS does not allow for negotiation and the client should immediately challenge the FTPS Server with the TLS/SSL handshake. The [Internet Assigned Numbers Authority](#) (IANA) officially designates port 990 as the FTPS control channel port and port 989 as the FTPS data channel port.

## SSL Certificates

Much like [https](#), but unlike [SFTP](#), FTPS servers must provide a [public key certificate](#). These certificates can be created using [Unix](#) tools such as [OpenSSL](#)'s `ssl-ca`.

This certificate must be signed by a [certificate authority](#), or the FTPS client will generate a warning stating that the certificate is not valid.

## The firewall problem

Because [FTP](#) is a port-hopping protocol (i.e. data channels use a random port chosen during the communication), many [firewalls](#) are designed to understand FTP protocol messages to determine what secondary data connections they need to allow. However, if the control connection is encrypted using TLS/SSL (or any other method for that matter), the firewall is not able to get the port numbers of the data connections from the control connection (since it is encrypted and the firewall cannot decrypt it). Therefore, in many firewalled networks, clear FTP connections will work while FTPS connections will either completely fail or require the use of passive mode (assuming all ports  $\geq 1024$  to the server are unfiltered).

## See also

- [List of file transfer protocols](#)
- [Comparison of FTP client software](#)
- [List of FTP server software](#)
- [Secure copy](#) (SCP)
- [SSH file transfer protocol](#) (SFTP)
- [FTP](#) (contains a section "FTP over SSH")
- [List of TCP and UDP port numbers](#)

## External links

- [RFC 4217](#) - Securing FTP with TLS
- [Overview of FTPS, and lists of clients, servers, and proxies supporting FTPS](#)