E

Encrypted File Transfer: SFTP, FTPS, HTTPS, and Safety

Compare SFTP, FTPS, HTTPS links, and file-level encryption, then use a practical checklist for identity, keys, integrity, and expiration.

PUBLISHER
Gridinsoft Support

UPDATED
Oct 8, 2026

READ TIME
3 min

CONTENTS

In this guide

01 SFTP vs. FTPS vs. HTTPS
02 Encryption in transit vs. at rest
03 Verify the server and recipient
04 Authentication and key management
05 How to send encrypted files safely
06 Sharing links and expiration
07 Common mistakes
08 When a transfer credential is exposed

Encrypted file transfer protects data while it moves between systems. Common options include SFTP over SSH, FTPS using TLS, HTTPS upload or download portals, and managed file-transfer services. File-level encryption can add protection before and after transport.

Encryption alone does not prove that the recipient is correct or that an account is authorized. Safe transfer also requires server authentication, access control, integrity, key management, logging, and deletion rules.

01 SFTP vs. FTPS vs. HTTPS

Method	Protection	Operational notes
SFTP	Runs as a file-transfer protocol over SSH	Usually one connection; verify the server host key
FTPS	Extends FTP with TLS	Uses separate control and data connections; firewall setup can be complex
HTTPS	Transfers through HTTP over TLS	Well suited to browsers, APIs, and expiring download links
File-level encryption	Encrypts the file itself	Can protect stored copies; keys must travel separately

SFTP is not “FTP over SSH,” and FTPS is not SFTP. The names are similar but the protocols and configuration differ.

02 Encryption in transit vs. at rest

Transport encryption protects the connection while data moves. It does not automatically protect temporary files, server storage, backups, email notifications, or a recipient’s downloaded copy. Encrypt sensitive files at rest when required and define when staging and shared copies are deleted.

03 Verify the server and recipient

For SFTP, verify the SSH host-key fingerprint through a trusted channel before accepting it; blindly accepting a changed key can enable interception. For FTPS and HTTPS, validate the hostname, certificate chain, and expiration. Do not disable certificate or host-key checks to make an automation job succeed.

Confirm the recipient and destination independently, especially after an email request to change account, hostname, or key.

04 Authentication and key management

Prefer unique service identities and narrowly scoped keys over shared passwords. Protect private keys with appropriate file permissions and a secret manager, rotate them, and revoke them when a user or integration is retired. MFA is valuable for interactive portals, while automated transfers need managed non-human identities and monitoring.

05 How to send encrypted files safely

1. Classify the file and confirm transfer is permitted.
2. Use an approved protocol with current TLS or SSH configuration.
3. Verify the server, recipient, and destination path.
4. Grant access only to the required file and expiration period.
5. Scan incoming and outgoing files where policy requires it.
6. Record a cryptographic hash or signed manifest when integrity must be verified.
7. Confirm receipt before deleting the source, then apply retention rules.

If file-level encryption is used, send the decryption password or key through a separate trusted channel—not in the same email or chat as the file. Confirm the recipient before disclosing it, and use a unique secret rather than reusing an account password.

06 Sharing links and expiration

An HTTPS link should use a strong, unguessable token, a short expiration, and the minimum required downloads. For sensitive data, require recipient authentication instead of treating possession of the link as permanent access. Avoid placing reusable secrets in URLs because browsers, proxies, and logs may retain them.

07 Common mistakes

- Using plain FTP or HTTP on an untrusted network.
- Emailing the decryption key with the encrypted attachment.
- Disabling certificate validation or ignoring a changed SSH host key.
- Sharing one account across people and integrations.
- Leaving uploaded files and public links active indefinitely.
- Assuming encryption prevents a compromised endpoint from reading the file.

08 When a transfer credential is exposed

Revoke the password, token, or key; review access and download logs; identify affected files; and notify the data owner. Replace credentials in automation through the approved secret store, not source code. If sensitive data may have been accessed, follow the organization’s incident, privacy, and legal procedures.

NEED MORE HELP?

We are here when the guide is not enough.

Visit the Gridinsoft Support Center for current product guidance or submit a support request with the details of your case.

Open this guide online at support.gridinsoft.com