

Using TLS to protect data

Recommendations for configuring TLS, managing certificates, supporting PQC and deploying secure services.

Introduction

This guidance is aimed at system administrators who are responsible for configuring servers or clients within their estate. It explains how to configure TLS securely and transition to post-quantum cryptography (PQC).

The guidance covers:

- transitioning TLS to use PQC, including recommended profiles for its use
- configuring TLS 1.3 and TLS 1.2 using recommended cryptographic profiles
- managing deprecated protocol versions and disabling known insecure TLS features
- using certificates with TLS
- deploying TLS for web and mail services
- testing TLS deployments

Key recommendations

- Use the [recommended profiles](#) when configuring TLS.
- Only TLS 1.3 provides protection against quantum attacks when configured correctly. You can use our recommended TLS 1.3 profiles now.
- If you use TLS 1.2, develop a plan to migrate to TLS 1.3 so you can adopt PQC.
- Disable TLS 1.1 and TLS 1.0. You must disable all versions of SSL.
- You must [disable TLS features known to be insecure](#).
- Keep servers and client software up to date. Implementation flaws can introduce vulnerabilities that may be exploited if security updates are not applied promptly.

About TLS

Transport Layer Security (TLS) is a protocol that provides security for digital communications between two parties. When a server and client communicate, well-configured TLS ensures that no third party can eavesdrop or tamper with any message.

Supported versions

TLS 1.3 is the latest version of TLS. It was designed to be more secure than earlier versions and is the only version that will receive new features in future, such as support for PQC.

TLS 1.2 is still considered to provide strong protection against classical attacks when configured with the recommended cryptographic profiles. However, it will not receive new features apart from critical security updates, as outlined in [RFC 9851](#).

If you use TLS 1.2, you should plan to transition to TLS 1.3 as part of your migration to PQC.

Deprecated protocols

The Secure Sockets Layer (SSL) protocol was the predecessor to TLS. All versions of SSL are deprecated and considered insecure. You must **not** be use them.

TLS 1.1 and 1.0 have also been deprecated by the IETF, as outlined in RFC 8996, and should not be used.

The NCSC recommends that government systems do **not** support deprecated versions of the protocol, and are upgraded to support TLS 1.3 instead.

Post-quantum cryptography in TLS

A cryptographically-relevant quantum computer would threaten the key exchange and digital signature algorithms currently used to secure TLS. The primary mitigation to this risk is to migrate to PQC, in line with the [broader NCSC advice on PQC migration](#).

Post-quantum key exchange in TLS

[RFC 10024](#) standardises three [post-quantum/traditional hybrid key agreement mechanisms](#) in TLS 1.3.

- [SecP256r1MLKEM768](#)
- [SepP384r1MLKEM1024](#)
- [X25519MLKEM768](#)

All three are suitable for protecting personal, enterprise and OFFICIAL-tier government information.

These mechanisms combine the post-quantum ML-KEM algorithm with a traditional elliptic curve algorithm. The NCSC recommends using hybrid schemes as an interim measure when migrating to PQC.

The IETF is developing standards for non-hybrid ML-KEM ([ML-KEM Post-Quantum Key Agreement for TLS 1.3](#)), which will standardise [MLKEM768](#) and [MLKEM1024](#) as options for TLS. The NCSC considers these algorithms acceptable for use in TLS, and they are already available in many common TLS implementations.

Post-quantum authentication in TLS

At present, the IETF has only standardised post-quantum algorithms for key exchange in TLS. Work is ongoing to standardise post-quantum authentication in TLS and the wider Web PKI, with a number of drafts at different levels of maturity. Organisations operating private PKIs may be able to adopt post-quantum authentication sooner than services relying on the public Web PKI, where standardisation and wider ecosystem adoption are still ongoing.

Once these standards have been finalised, and mature implementations are available in commercial implementations, this guidance will be updated to include recommendations for their use.

Using cryptographically agile deployments, where it is easier to support alternative cryptographic algorithms, will make these and future transitions easier.

TLS 1.3 configuration

TLS 1.3 introduced some major changes. It uses different cipher suite definitions to earlier TLS versions, and has different configuration options. Some implementations separate TLS 1.3 configuration from previous TLS versions. Please consult your implementation documentation to determine how to configure options specific to TLS 1.3.

The recommended profiles include both post-quantum and traditional algorithms to support the transition to PQC. The recommended symmetric cipher suites are unchanged.

The key exchange algorithms included in the recommended profiles are not the only ones the NCSC considers acceptable. All of the post-quantum key exchange algorithms currently specified for use with TLS 1.3 by the IETF offer appropriate protection for personal, enterprise and OFFICIAL-tier government information.

Recommended post-quantum cryptography profiles for TLS 1.3

Hybrid post-quantum / traditional profile		
Key Exchange	X25519MLKEM768 or SecP256r1MLKEM768	Hybrid of ECDHE on Curve 25519 and MLKEM 768 or Hybrid of ECDHE on the P-256 curve and MLKEM 768
Authentication	ecdsa_secp256r1_sha256 or rsa_pss_rsae_sha256 or rsa_pss_pss_sha256	ECDSA with SHA256 on P-256 curve or RSA signatures with 2048-bit modulus and SHA256 digests
Cipher Suite	TLS_AES_128_GCM_SHA256	AES with 128-bit key using GCM for encryption SHA256 for the HKDF algorithm

Pure post-quantum profile		
Key Exchange	secp256r1	MLKEM 768
Authentication	ecdsa_secp256r1_sha256 or rsa_pss_rsae_sha256 or rsa_pss_pss_sha256	ECDSA with SHA256 on P-256 curve or RSA signatures with 2048-bit modulus and SHA256 digests
Cipher Suite	TLS_AES_128_GCM_SHA256	AES with 128-bit key using GCM for encryption SHA256 for the HKDF algorithm

Recommended traditional cryptography profiles for TLS 1.3

Recommended traditional cryptography profiles		
Key Exchange	secp256r1	ECDHE on the P-256 curve
Authentication	ecdsa_secp256r1_sha256 or rsa_pss_rsae_sha256 or rsa_pss_pss_sha256	ECDSA with SHA256 on P-256 curve or RSA signatures with 2048-bit modulus and SHA256 digests
Cipher Suite	TLS_AES_128_GCM_SHA256	AES with 128-bit key using GCM for encryption SHA256 for the HKDF algorithm

OpenSSL Configuration

The way preferred algorithms and cipher suites are configured varies between TLS platforms. As an example, for platforms built around the OpenSSL library, the following recommended configuration string for TLS 1.3 servers will enable a set of key exchange algorithms (or ‘supported groups’) that aims to optimise security, compatibility and performance:

`*X25519MLKEM768:SecP256r1MLKEM768:MLKEM768/SecP384r1MLKEM1024:MLKEM1024/*secp256r1:secp384r1`

Client configuration for TLS 1.3

TLS clients should offer key shares and cipher suites that are most likely to be accepted by the server. The configuration of TLS clients is often outside the control of the administrator, but where it is possible to configure clients (for example within a privately hosted enterprise network), the NCSC recommends offering key shares for X25519MLKEM768 and secp256r1.

Using the recommended profiles for PQC migration

To support PQC in TLS, you should:

- enable TLS 1.3 and prefer it over TLS 1.2 where possible

- enable at least one of the recommended post-quantum key exchange algorithms as soon as your implementation supports them
- enable the recommended traditional key exchange algorithms where you still require compatibility with systems that do not yet support PQC

In most cases, it will be appropriate to combine these activities by adopting TLS 1.3 and enabling and preferring the recommended post-quantum key exchange algorithms as a single migration activity.

Migrating the authentication methods used by TLS 1.3 to post-quantum alternatives will depend on the transition of the PKI infrastructure they rely on, and will need to wait until the relevant standards have been confirmed.

For internet-facing servers and clients, maintaining compatibility with systems that do not yet support PQC is important. As a result, support for traditional cryptography cannot be disabled until much later in the overall PQC migration.

TLS 1.2 configuration

Where client and server implementations support AES-GCM, use one of the recommended profiles below. These both provide equivalent protection.

If AES-GCM is not supported, then the compatibility profile for TLS 1.2.

Recommended Profiles for TLS 1.2

Key Exchange	secp256r1	ECDHE on the P-256 curve
Authentication	SHA256 and ECDSA using secp256r1	ECDSA with SHA256 on P-256 curve
Cipher Suite	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	ECDSA for key exchange ECDSA for authentication AES with 128-bit key using GCM for encryption SHA256 for the PRF

Key Exchange	secp256r1	ECDHE on the P-256 curve
Authentication	SHA256 and RSASSA-PKCS1-v1_5 with 2048 bit modulus	RSA signatures with 2048-bit modulus and SHA256 digests
Cipher Suite	TLS_ECDHA_RSA_WITH_AES_128_GCM_SHA256	ECDHE for key exchange RSA for authentication AES with 128-bit key using GCM for encryption SHA256 for the PRF

Compatibility Profile for TLS 1.2

Key Exchange	secp256r1	ECDHE on the P-256 curve
Authentication	SHA256 and RSASSA-PKCS1-v1_5 with 2048 bit modulus	RSA signatures with 2048-bit modulus and SHA256 digests
Cipher Suite	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDHE for key exchange RSA for authentication AES with 128-bit key using CBC for encryption SHA256 for the MAC

Legacy versions

Legacy Profiles for TLS 1.1 and 1.0

TLS versions 1.1 and 1.0 do not support current authenticated encryption cipher suites and have other known vulnerabilities. These versions have been formally deprecated by the IETF, as outlined in RFC 8996, and should no longer be used.

A time-bounded migration plan should be put in place for any government systems using deprecated TLS versions, or cipher suites. Where this is not possible, system owners should consult the NCSC.

The following legacy profiles are defined only for cases where TLS versions 1.1 and 1.0 cannot be disabled. In all other circumstances, the recommended TLS 1.2 or 1.3 profiles should be used, and TLS 1.0 and TLS 1.1 must be disabled.

Key Exchange	ECDHE using P-256 curve
Authentication	RSA signatures with 2048-bit modulus and SHA256 digests
Encryption	AES with 128-bit key using CBC
MAC	SHA1
Cipher Suite	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

Key Exchange	DH Group 14 (2048-bit modulus)
Authentication	RSA signatures with 2048-bit modulus and SHA256 digests
Encryption	AES with 128-bit key using CBC
MAC	SHA1
Cipher Suite	TLS_DHE_RSA_WITH_AES_128_CBC_SHA

Third-Party services

When a TLS service is hosted on the infrastructure of a third-party provider, such as a content delivery network (CDN), the TLS configuration may not be under the direct control of the data owner. Service providers will usually allow the data owner to choose a policy that specifies which TLS versions and cipher suites are permitted.

The data owner should choose the most restrictive policy available. This policy should enable TLS 1.3 and where required TLS 1.2, and should include the recommended profiles described in this guidance.

Avoiding security weaknesses in older versions

Older TLS versions include features and cryptographic components that are now known to provide weak security. In most cases, these features are only used in a connection when both the client and server support them.

Using up-to-date software reduces the likelihood of insecure features being used. To avoid some of the most common vulnerabilities, you must **disable the following features and components** where they are present:

- insecure TLS renegotiation – either by disabling renegotiation, or enabling the Renegotiation Indication Extension, as described in [RFC 5746](#)
- insecure TLS protocol downgrade – by supporting the Fallback Signaling Cipher Suite Value, as described in [RFC 7507](#), only where TLS 1.1 or earlier versions are supported
- TLS record compression – as described in [RFC 5246, section 6.2.2](#)
- EXPORT ciphers and export key generation – as described in [RFC 2246, section 6.3.1](#)

You should also disable support for any deprecated SSL and TLS versions that are not required.

Using TLS 1.3 0-RTT

TLS 1.3 provides an optional zero round trip feature, known as 0-RTT, which allows clients to send data before the full TLS handshake has completed. Unlike most data exchanged in a TLS session, 0-RTT data can be [replayed](#) by an attacker. If this

risk is not addressed by the application using TLS, it can lead to security vulnerabilities.

Some application protocols built on TLS 1.3 include mitigations for attempted replay attacks, and can safely use 0-RTT mode. You should only enable 0-RTT when you are satisfied that the software and application protocols in use to provide appropriate mitigations.

Using Certificates with TLS

TLS uses X.509 certificates to cryptographically verify the identity of one or both communicating parties. A secure TLS connection can be established when the connecting party trusts the certificate presented by the service and can verify that it was issued by a trusted Certificate Authority (CA).

Choosing a CA

In many cases a TLS service will need to be used and trusted by unknown clients, for example members of the public. In this case the service must use a certificate from a public CA in the Web PKI. The NCSC provides guidance on [provisioning and managing certificates in the web PKI](#).

Many public CAs are available, with differing levels of trust across client trust stores, technical support offerings, and certificate issuance and management options. When selecting a CA, you should ensure that it is trusted by the clients that need to access your service and that it meets your requirements.

If you are hosting your service in the cloud, many cloud providers offer CA services and host their own CAs. Using these cloud-native CA services can simplify integration and automate certificate management, provided the CA offers sufficient security properties for your purposes. The NCSC's [cloud security guidance](#) will help you make this assessment.

If your organisation manages its own internal CA, and the service is only accessed by managed clients that trust it, it may be appropriate to use certificates issued by the internal CA. See the NCSC's guidance on [designing and building a privately hosted public key infrastructure \(PKI\)](#).

Recommended certificate parameters

When requesting or renewing a certificate, you will need to choose parameters for your private key and certificate signing request (CSR). Whether you're using a public or a private CA, the NCSC recommends using the following parameters for generation of ECC and RSA keys:

- ECDSA-256 with SHA256 on the P-256 curve
- 2048-bit RSA with SHA256

Planning for PQC

International standards for post-quantum authentication mechanisms and certificates in TLS are still being developed. Until those standards are available, you should continue using ECDSA and RSA. This recommendation will be updated to prefer post-quantum authentication mechanisms once they have been standardised.

When issuing certificates for ECDSA and RSA, you should consider your organisation's [PQC migration timelines](#). Avoid issuing certificates with validity periods that extend beyond the point at which you expect to complete your migration from traditional cryptography. You should also identify any existing certificates that may need to be replaced as part of that transition.

Deploying TLS for web servers

When deploying TLS for web services, you should use the recommended TLS profiles described in this guidance. In addition, you should use the following HTTPS security features:

- serve websites and services only using HTTPS
- redirect unencrypted HTTP requests to HTTPS
- use HTTP Strict Transport Security (HSTS) and preload your HSTS policy to ensure all connections to your web server use HTTPS
- use the [upgrade-insecure-requests directive](#) in your [Content Security Policy](#), to ensure site content, including third party content, is loaded using HTTPS
- mark cookies as “secure” so they are only transmitted over TLS-protected connections

For information about HSTS preloading, see <https://hstspreload.org/>

Deploying TLS for mail servers

Deploying TLS for email services can be more complex than for other scenarios. If configured incorrectly, mail may be sent unencrypted, or may silently fail to be delivered.

Mail transfer between servers

When handling connections between Mail Transfer Agents (MTAs) using SMTP STARTTLS on port 25, compatibility with other MTAs is required to ensure that mail is not sent unencrypted due to a TLS negotiation failure. To reduce the chance of incompatibility, use the recommended TLS 1.3 and 1.2 profiles and the TLS 1.2 compatibility profile.

You should inspect your mail server logs to identify connections that are falling back to unencrypted delivery because of TLS configuration issues.

Connections from end users

For handling connections from end users via Mail User Agents (MUAs) using protocols such as IMAP, POP3 and SMTP Submission, mail servers should use the recommended TLS 1.3 and 1.2 profiles.

As outlined in [RFC 8314](#), STARTTLS is no longer recommended for communications between end users and mail services. Instead, use 'Implicit TLS', where the TLS session starts immediately after connection establishment. For example, IMAP services should use IMAPS on TCP port 993 rather than STARTTLS on the IMAP TCP port 143.

In addition to this guidance, you should refer to the [NCSC's guidance on email security and anti-spoofing](#).

Additional protections

You may also wish to deploy the following standards to improve the security and visibility of TLS-protected email delivery:

Mail Transfer Agent Strict Transport Security (MTA-STS)

- Deploy [MTA-STS](#) to require supporting MTAs to deliver email using TLS. Start by deploying in *testing* mode, before moving to *enforce* mode to reduce the risk of mail delivery problems caused by configuration errors.

SMTP TLS Reporting (TLS-RPT)

- Deploy TLS-RPT, following the [NCSC's email security and anti-spoofing guidance](#), to allow supporting MTAs to report TLS-related issues to mail server administrators. This is strongly recommended when using MTA-STS, as it can help identify problems that may prevent email delivery.

Certificate validation considerations

MTAs that support MTA-STS perform stricter certificate validation.

When MTA-STS is enabled:

- the mail server certificate is expected to match against the host name specified in the relevant MX record
 - the certificate must chain to a CA trusted by the sending system
-

Testing web and mail servers

Given the wide range of configuration options available for TLS, you should regularly test your web and mail server configurations. External Attack Surface Management (EASM) products can help identify common TLS configuration problems. For further guidance, see the NCSC's [EASM buyers guide](#).

EASM products are not a replacement for skilled penetration testing of your services. However, identifying and addressing common configuration issues before a penetration test allows testers to focus on more complex and service-specific vulnerabilities.

External services can test whether your organisation is able to receive email securely. However, they cannot determine whether your systems send email securely without your involvement. To test outbound email security, you will need to send email to a suitable testing service.

Using TLS correctly is an important part of securing web and mail services, but it should be noted that it is only one aspect and should be combined with other measures as part of a broader security strategy.

PUBLISHED

15 October 2026

REVIEWED

15 October 2026

VERSION

1.2

WRITTEN FOR

Cyber security professionals

Large organisations

Public sector

Small & medium sized organisations