



Post Quantum Cryptography for MACsec with EAP TLS

- [Overview of PQC for MACsec with EAP-TLS, on page 1](#)
- [Key elements of PQC, on page 2](#)
- [PQC implementation in EAP-TLS, on page 3](#)
- [Prerequisites, on page 4](#)
- [Restrictions, on page 4](#)
- [Configuration matrix for MACsec with EAP-TLS, on page 5](#)
- [Workflow to set up PQC for MACsec with EAP-TLS, on page 6](#)
- [Configure EAP-TLS 1.3, on page 6](#)
- [Configure PQC for EAP-TLS sessions, on page 7](#)
- [Verify PQC for EAP-TLS sessions, on page 7](#)

Overview of PQC for MACsec with EAP-TLS

Post Quantum Cryptography (PQC) describes cryptographic algorithms that remain secure against attacks from quantum computers. Quantum computers can break many classical encryption methods. Widely used encryption methods—such as Diffie-Hellman (DH) and Elliptic Curve Diffie-Hellman (ECDH)—are vulnerable as quantum computing technology advances.

Implementing quantum-safe encryption now is critical for the long-term confidentiality and integrity of sensitive communications, particularly for government, military, and financial sectors.

PQC for MACsec with EAP-TLS

MACsec encryption with EAP-TLS authentication establishes secure communication between switches by using certificate-based mutual authentication to derive keys for MACsec encryption. This process ensures that data traffic on the MACsec link is encrypted and decrypted and maintains integrity and confidentiality throughout the session.

Post Quantum Cryptography for MACsec with EAP-TLS provides line-rate Layer 2 encryption with quantum-safe mutual authentication. This section provides the configuration information for MACsec with EAP-TLS sessions.

Key elements of PQC

PQC is implemented by using the algorithms available as part of Module-lattice based Cryptography. Module-lattice based Cryptography is a quantum-resistant public-key cryptography that uses structured, high-dimensional grid points called lattices to secure data. This approach combines module-lattice based Cryptography (ML-KEM) with an elliptic curve function such as p256.

This combination provides enhanced security by protecting against both classical and quantum attacks while maintaining compatibility and performance with existing cryptographic protocols.

This table lists the supported algorithms and the areas in which each algorithm can be used.

Table 1: Key elements of PQC

Algorithm	What this algorithm means	Security level
• mlkem512 • mlkem768 • mlkem1024	ML-KEM is a PQC algorithm based on module-lattice cryptography, providing resistance against quantum attacks.	The security level of this implementation is balanced and is best suited for deployments where performance metrics are critical.
• p256_mlkem512 • p384_mlkem768 • p521_mlkem1024	ML-KEM is a PQC algorithm based on module-lattice cryptography, providing resistance against quantum attacks. p256, p384, and p521 refer to the traditional (classical) component. The algorithm uses Elliptic Curve Diffie-Hellman (ECDH) with the NIST P-256, P-384, or P-521 curves to maintain security against current non-quantum threats.	The security level of this implementation is the highest and is best suited for deployments where maximum security is needed.
• x25519_mlkem512 • x448_mlkem768 • X25519MLKEM76	ML-KEM is a PQC algorithm based on module-lattice cryptography, providing resistance against quantum attacks. X448 and X25519 are widely used classical elliptic curve Diffie-Hellman key exchange algorithms. This hybrid PQC key exchange algorithm is used to secure key exchanges against quantum and classical threats in Cisco's PQC-enabled network protocols.	The security level of this implementation is the highest and is best suited for deployments where maximum security is needed.

Algorithm	What this algorithm means	Security level
• SecP256r1MLKEM768 • SecP384r1MLKEM1024	secp384r1 and secp521r1 are standard, classical Elliptic Curve Diffie-Hellman (ECDH) groups used for key exchange during the TLS 1.3 handshake. ML-KEM is a PQC algorithm based on module-lattice cryptography, providing resistance against quantum attacks.	The security level of this implementation is good and is best suited for most deployments.
• secp256r1 • secp384r1 • secp521r1	Secp256r1, secp384r1, and secp521r1 are standard, classical ECDH groups used for key exchange during the TLS 1.3 handshake. These are the standard, non-quantum-resistant cryptographic curves that the system uses to ensure interoperability and maintain security when PQC is not required or not supported by the peer.	If a PQC-compliant handshake cannot be established (for example, when the pqc-type is set to all or non-pqc), the system falls back to using these classic groups. This approach is best suited for deployments where PQC is not chosen.

PQC implementation in EAP-TLS

PQC in Extensible Authentication Protocol Transport Layer Security (EAP-TLS) is implemented through negotiated key-exchange configuration. TLS 1.3 is mandatory for this implementation. This implementation secures Ethernet traffic through MACsec encryption. It combines IEEE 802.1X port-based authentication and EAP-TLS certificates for mutual authentication and automated key derivation.

MACsec encryption with EAP-TLS secures switch-to-switch communication. Certificate-based mutual authentication derives cryptographic keys, which are managed by the MACsec Key Agreement (MKA) protocol to ensure data confidentiality and integrity across Ethernet interfaces.

MACsec encryption uses EAP-TLS authentication to establish secure communication between switches. Certificate-based mutual authentication is leveraged to derive keys for MACsec encryption.

Summary

The key components involved in the process are

- Switches (authenticator/supplicant): Systems that perform MACsec encryption and participate in 802.1X authentication, acting as either the authenticator (facilitates authentication) or the supplicant (seeks authentication).
- Authentication server: An entity that provides authentication services to an authenticator, verifying supplicant credentials and facilitating EAP-TLS communication.

- Certificate Authority (CA) server: Issues and manages digital certificates used for mutual authentication in EAP-TLS.
- EAP-TLS (Extensible Authentication Protocol-Transport Layer Security): The authentication method used for mutual authentication between the authentication server and the client (supplicant) using certificates.
- Master Session Key (MSK): A cryptographic key generated upon successful EAP-TLS authentication.
- Connectivity Association Key (CAK): Derived from the MSK, this key is used by the MACsec Key Agreement (MKA) protocol. CAK is used to derive Secure Association Key (SAK), which performs the actual encryption and authentication of data packets on the link.
- Connectivity Association Key Name (CKN): Derived from the EAP session ID. This name identifies the CAK.

Workflow

These stages describe how MACsec encryption using EAP-TLS authentication works:

1. Initiation: A supplicant switch initiates 802.1X port-based authentication on a physical Ethernet interface with an authenticator switch.
2. Mutual authentication (EAP-TLS): The authentication server and the supplicant switch perform mutual authentication using digital certificates via the EAP-TLS method. This requires both devices to have valid certificates issued by a trusted Certificate Authority.
3. Master session key generation: Upon successful EAP-TLS authentication, a Master Session Key (MSK) is generated.
4. Key derivation: The MSK is then used to derive the Connectivity Association Key (CAK), and the Connectivity Association Key Name (CKN) is derived from the EAP session ID.
5. MACsec Key Agreement (MKA): The derived CAK and CKN are utilized by the MKA protocol to establish and maintain secure MACsec encryption between the switches on the interface.

Result

This process enables robust MACsec encryption between two switches, ensuring data confidentiality and integrity on Ethernet interfaces through secure, certificate-based authentication and automated key management.

Prerequisites

This list specifies the prerequisites before you configure PQC for MACSec with EAP-TLS.

- Ensure MACsec is enabled on the authenticator's (the switch that facilitates authentication) interface and ensure that the supplicant (the switch that seeks authentication) is capable of MACsec.

Restrictions

This list provides the limitations for configuring PQC in MACsec with EAP-TLS.

- In Cisco IOS XE 26.1.1, PQC for EAP-TLS is not supported in Federal Information Processing Standards (FIPS) mode.
- PQC for EAP-TLS is not supported with TLS versions 1.0 and 1.2.
- Certificate Revocation List (CRL) revocation is not supported for MACsec with EAP-TLS.
- PQC is supported with only local authentication for MACsec EAP -TLS.

Configuration matrix for MACsec with EAP-TLS

The protocol and TLS version matrix lists the combinations required for PQC to operate as expected for EAP-TLS. In this matrix, both device 1 and device 2 are switches.

Device 1 protocol and version	Device 2 protocol and version	Negotiated TLS version	Authentication status
EAP-TLS – 1.3	EAP-TLS – 1.3	1.3	Authentication successful
EAP-TLS – all	EAP-TLS – all	1.3	Authentication successful
EAP-TLS-1.2	EAP-TLS-all	1.2	Authentication successful
EAP-TLS – 1.3	EAP-TLS – 1.2		Authentication failure
EAP-TLS – 1.0	EAP-TLS – 1.3		Authentication failure

Supported key-exchange type for the algorithms

The table specifies the key-exchange type supported for each algorithm. The table lists these algorithms used to verify whether PQC is supported on your platform.

- **All:** all PQC, non-PQC and hybrid algorithms are supported.
- **Hybrid:** combined PQC and non-PQC algorithms are supported.
- **Non-PQC:** these are the classic cryptographic algorithms without PQC.
- **PQC:** PQC algorithms.

Version or key-exchange-type	Classic	PQC	Hybrid	All
1.3	Supported	Supported	Supported	Supported
1.2	Supported	Not supported	Not supported	Supported
1.0	Supported	Not supported	Not supported	Supported

Workflow to set up PQC for MACsec with EAP-TLS

This section describes how an authenticator and a supplicant can complete EAP-TLS authentication using ML-KEM for the TLS key exchange. This process results in the establishment of a secure MACsec session.

The steps to set up this configuration are:

1. Enable EAP-TLS with support for PQC ML-KEM on the supplicant as well as the authenticator.
2. Install and trust valid client and server certificates on the respective devices.
3. Connect the supplicant to the authenticator's MACsec-enabled port.
4. Initiate the authentication process by sending traffic from the supplicant or enabling the interface.
5. Verify PQC for EAP-TLS sessions.

Configure EAP-TLS 1.3

Follow these steps to configure TLS 1.3. If the peer device supports TLS version 1.3, authentication succeeds through EAP-TLS 1.3; otherwise, authentication fails.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
Switch# configure terminal
```

Step 2 Use the **access-session tls-version version** command to configure the TLS 1.3 version.

Example:

```
Switch (config)# access-session tls-version <1.3>
```

Sample configuration

```
Switch (config)# access-session tls-version ?
  1.0 Version 1.0
  1.2 Version 1.2
  1.3 Version 1.3
  all All TLS version
```

By default, **all** is configured on a device. When both the devices support TLS 1.3, TLS 1.3 negotiation is successful. Else, an older version of TLS version is enabled. In this scenario, PQC does not work.

Configure PQC for EAP-TLS sessions

Follow these steps to enable PQC for EAP-TLS sessions.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
Switch# configure terminal
```

Step 2 Use the **access-session pqc-type pqc** to configure the PQC type.

Example:

```
Switch (config)# access-session pqc-type pqc
```

Note

If a device does not support PQC, the system displays this warning message:

```
Switch(config)# access-session pqc-type pqc
Warning! This platform only supports non-pqc method
```

Sample configuration

```
Switch(config)# access-session pqc-type ?
all          All pqc, non-pqc and hybrid algorithms will be supported
hybrid       Combined PQC and NON-PQC algorithms (Key derived with combination of both)
non-pqc      Classic cryptographic algorithms
pqc          Post-Quantum Cryptographic algorithms
```

Verify PQC for EAP-TLS sessions

Use these commands to verify whether PQC implementation is successful for EAP-TLS sessions.

Command	Purpose	What to look out for
Device(config)# access-session tls-version ?	Verify the TLS version	The TLS version should read 1.3 for this functionality to work.
show running-config all i pqc-type	Verify the configured PQC type	
show dot1x interface <intf> detail	Verify the EAP profile is set up and the EAP method	The EAP method should read TLS.

