

IKEv2 and IPsec PFS Post Quantum Cryptography

Table 1: Feature History Table

Feature name	Release information	Feature description
IKEv2 and IPsec PFS Post Quantum Cryptography	Release 26.2.1	Cisco IoT routers can combine a traditional Diffie-Hellman key exchange with the NIST-standardized Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM) to establish keys for IKEv2/IPsec sessions. This hybrid approach helps protect recorded VPN traffic against future harvest-now, decrypt-later attacks.

- IKEv2 and IPsec PFS Post Quantum Cryptography, on page 1
- Benefits of PQC, on page 2
- Prerequisites, on page 2
- Limitations, on page 2
- ML-KEM algorithm and parameter sets, on page 3
- PFS negotiation behavior, on page 4
- Configure an IKEv2 proposal for PQC, on page 4
- Configure Perfect Forward Secrecy with PQC, on page 6
- Verify the PQC hybrid key exchange, on page 7
- Troubleshooting PQC hybrid key exchange, on page 8

IKEv2 and IPsec PFS Post Quantum Cryptography

Quantum-Safe Encryption refers to cryptographic techniques designed to protect network communications against attacks by computers and future cryptographically relevant quantum computers.

Post-Quantum Cryptography (PQC) is part of this broader security approach. PQC includes quantum-resistant algorithms for functions such as key establishment and digital signatures. These algorithms help mitigate harvest-now, decrypt-later attacks, in which an attacker records encrypted traffic and attempts to decrypt it later using a sufficiently capable quantum computer.

PQC does not replace existing public-key algorithms. Instead, it works alongside them to create a quantum-safe hybrid key exchange.

The Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM) is a National Institute of Standards and Technology (NIST)-standardized post-quantum key-encapsulation mechanism. ML-KEM enables two peers to establish a shared secret that can be used to derive cryptographic session keys.

On supported Cisco routers, ML-KEM is combined with a traditional key-exchange algorithm to provide a hybrid key establishment for IKEv2/IPsec Perfect Forward Secrecy (PFS). The hybrid approach protects the negotiated session as long as at least one of the constituent key-establishment mechanisms remains secure.

For more information on PQC on Cisco routers with IKEv2 sessions, see Security and VPN Configuration Guide.

For more information on Quantum-Safe Encryption and the related protocols, see: https://www.cisco.com/c/en/us/td/docs/routers/ios/config/17-x/sec-vpn/b-security-vpn/m-sec-cfg-quantum-encryption-ppk.html.



Note Postquantum preshared keys (PPKs), including dynamic PPKs obtained through the Secure Key Integration Protocol (SKIP), can be used with ML-KEM-based hybrid key establishment.

Benefits of PQC

You can use the PQC feature to strengthen the security of your IPsec and IKEv2 connections with quantum-safe, future-proof protection for traffic between Cisco IoT routers.

The PQC hybrid key exchange provides these benefits:

- Future-proof security: Protects traffic against future quantum-computer-based attacks.
- Defense in depth: Combines a classical DH algorithm with ML-KEM, so the session stays secure even if one algorithm is broken.
- Smooth migration: Supports phased migration in existing networks through the optional fallback keyword.

Prerequisites

Before you configure the PQC hybrid key exchange, ensure that the following requirements are met:

- Both peers support PQC, or the optional keyword is used to allow fallback during migration.
- IKEv2/IPsec is configured and operational between the peers.

Limitations

The limitations with respect to using PQC are:

- IKEv2 always uses hybrid key exchange: a combination of Diffie-Hellman (DH) and ML-KEM algorithms. Pure-PQC key exchange is not supported for IKEv2.

- The encapsulation keys and ciphertexts used by ML-KEM-768 and ML-KEM-1024 may cause IKEv2 messages to exceed typical path maximum transmission units (MTUs), resulting in IP fragmentation and potential packet drops. To avoid IP fragmentation, enable IKEv2 fragmentation and set the IKEv2 fragmentation MTU to 1400 bytes.
- This feature provides post-quantum key establishment only. It does not provide post-quantum peer authentication. IKEv2 peers continue to use the configured authentication method, such as a preshared key or certificate-based authentication.
- ML-KEM establishes shared keying material; it does not encrypt user traffic. The encryption algorithm configured in the IPsec transform set, such as AES-CBC or AES-GCM, continues to encrypt the user traffic.

Note License, platform, and software support: This feature does not require an additional feature license. ML-KEM-based quantum-safe key establishment is supported on all Cisco IoT routers with Cisco IOS XE Release 26.2.1.

Phased migration in Hub-and-Spoke deployments

If an older responder (hub) receives a PQC proposal from upgraded spokes (initiators), the negotiation fails.

To avoid this, it is recommended that you always upgrade the hub first and configure **optional** in the hub's IKEv2 proposal.

When any issue occurs, the responder logs the following errors:

IKEv2-INTERNAL: Transform type 6 invalid.

IKEv2-ERROR: Failed to parse the packet: The peer's proposal is invalid.

ML-KEM algorithm and parameter sets

Cisco IoT routers support the three National Institute of Standards and Technology (NIST)-standardized ML-KEM parameter sets: ML-KEM-512, ML-KEM-768, and ML-KEM-1024. During the key exchange, the peers exchange an encapsulation key (public key) and a ciphertext.

Table 2: ML-KEM parameter sets (sizes in bytes)

Parameter	ML-KEM-512	ML-KEM-768	ML-KEM-1024
Encapsulation key size (public key)	800	1184	1568
Decapsulation key size (private key)	1632	2400	3168
Shared secret key size	32	32	32
Ciphertext size	768	1088	1568

PFS negotiation behavior

Table 3: PFS negotiation behavior for Cisco IOS XE to Cisco IOS XE devices

Initiator/ Responder	no set pfs	set pfs	set pfs <group>	set pfs <group> pqc <mlkem>
set pfs	UP (ML-KEM inherit)	UP (ML-KEM inherit)	UP (DH+ML-KEM inherit)	UP (DH+ML-KEM inherit)
set pfs <*group*>	UP (DH)	DOWN	UP (DH)	DOWN
set pfs <*group*> **pqc** <*mlkem*>	UP (DH+ML-KEM)	UP (DH+ML-KEM inherit)	UP (DH+ML-KEM)	UP (DH+ML-KEM)

Note During initial session establishment, `CHILD_SA` output might display `PFS: N`. After the `CHILD_SA` rekeys, verify the negotiated ML-KEM and DH PFS settings.

Configure an IKEv2 proposal for PQC

Follow these steps to enable ML-KEM algorithm in an IKEv2 proposal.

Procedure

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

Example:

```
Router# configure terminal
```

Step 2 Use the **crypto ikev2 fragmentation mtu** *mtu-size* command to enable IKEv2 fragmentation and set the IKEv2 fragmentation MTU to 1400 bytes.

Example:

```
Router(config)# crypto ikev2 fragmentation mtu 1400
```

Step 3 Use the **crypto ikev2 proposal** *name* command to create or edit an IKEv2 proposal.

Example:

```
Router(config)# crypto ikev2 proposal prop1
```

Step 4 Use the **encryption** *aes-cbc-key* command to configure AES key in CBC mode.

Example:

```
Router(config-ikev2-proposal)# encryption aes-cbc-256
```

a) Use the **integrity** *integrity-algorithm* command to configure the IKEv2 integrity algorithm.

Example:

```
Router(config-ikev2-proposal)# integrity sha512
```

b) Use the **group** *group-number* command to configure Diffie-Hellman group for classical key exchange used with ML-KEM.

Example:

```
Router(config-ikev2-proposal)# group 19
```

Step 5 Use the **pqc mlkem1024 mlkem768 mlkem512 optional** command to enable one or more ML-KEM algorithm parameter sets.

Example:

```
Router(config-ikev2-proposal)# pqc mlkem1024 mlkem768 mlkem512 optional
```

Note The proposal enables **pqc mlkem768 optional** by default. The **optional** keyword allows the SA to fall back to a classical (non-PQC) negotiation if the peer does not support PQC, which supports coexistence during phased migration.

Step 6 Use these commands to create the IKEv2 policy and attach the proposal:

a) Use the **crypto ikev2 policy** *policy* command to create the IKEv2 policy.

Example:

```
Router(config)# crypto ikev2 policy pol1
```

b) Use the **match fvrf any** command to apply the IKEv2 policy to negotiations through any front-door VRF (FVRF).

Example:

```
Router(config-ikev2-policy)# match fvrf any
```

c) Use the **proposal** *proposal-name* command to add the PQC-enabled proposal to the IKEv2 policy.

Example:

```
Router(config-ikev2-policy)# proposal prop1
```

The **proposal** *proposal* command makes the PQC-enabled proposal available during IKEv2 negotiation for sessions that match this policy.

Step 7 Use these commands to associate the existing IKEv2 profile with the IPsec profile:

a) Use the **crypto ipsec profile** *profile-name* command to create an IPsec profile.

Example:

```
Router(config)# crypto ipsec profile gen
```

b) Use the **set ikev2-profile** *ikev2-profile-name* command to associate an existing IKEv2 profile with the IPsec profile.

Example:

```
Router(ipsec-profile)# set ikev2-profile prof1
```

Note The profile and IPsec transform set can already exist, if not you can create it using the create IPsec profile option given in step 7.

Step 8 Use these commands to apply the IPsec profile to the tunnel:

a) Use the **interface tunnel** *tunnel-interface-number* command to create a tunnel interface.

Example:

```
Router(config)# interface tunnel 1
```

b) Use the **tunnel protection ipsec profile** *ipsec-profile-name* command to protect the tunnel with IPsec profile.

Example:

```
Router(config-if)# tunnel protection ipsec profile gen
```

Step 9 Use the **end** command to return to privileged EXEC mode.

Example:

```
Router(config-if)# end
```

Configure Perfect Forward Secrecy with PQC

Perfect Forward Secrecy (PFS) adds extra randomness to derive session keys during each key exchange. With PQC, the key exchange can use DH, ML-KEM, or DH + ML-KEM. PFS is optional and can be disabled.

Follow these steps to configure PFS with PQC.

Procedure

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

Example:

```
Router# configure terminal
```

Step 2 Use the **crypto ipsec profile** *name* command to enter the IPsec profile.

Example:

```
Router(config)# crypto ipsec profile P1
```

Step 3 Use the **set pfs** *group* **pqc** *mlkem-algorithm* command to enable PFS, specify a DH group, and an ML-KEM algorithm, or both.

Example:

```
Router(ipsec-profile)# set pfs group19 pqc mlkem1024
```

Note When **set pfs** is used:

- Without PQC: Starting from Cisco IOS XE Release 26.2.1 onwards, the DH group is inherited from the IKEv2 SA.
- With PQC: The DH group is inherited from the IKEv2 SA. Exchanges are optimized for Cisco IOS XE to Cisco IOS XE deployments using an ML-KEM-only IPsec Child Security Association exchange.

Verify the PQC hybrid key exchange

Follow these steps to verify that the PQC hybrid key exchange is active.

Procedure

Step 1 Use the **show crypto ikev2 sa detail** command to verify the IKEv2 SA and the PQC key exchange algorithm.

Example:

```
Router# show crypto ikev2 sa detail

IPv4 Crypto IKEv2 SA

Tunnel-id Local              Remote             fvrf/ivrf   Status
1         10.0.149.203/500   10.0.149.217/500   none/none   READY
Encr: AES-CBC, keysize: 256, PRF: SHA512, Hash: SHA512, DH Grp:19, Auth sign: PSK, Auth verify: PSK
PQC Key Exchange: ML-KEM-1024
...
Quantum-safe Encryption using PQC: ML-KEM-1024
PEER TYPE: IOS-XE
```

Step 2 Use the **show crypto ipsec sa** command to verify the IPsec SA. The PQC key exchange appears after the first IPsec SA rekey.

Example:

```
Router# show crypto ipsec sa
...
PFS (Y/N): Y, DH group: group19, PQC Key Exchange: ML-KEM-1024
```

Note The PQC key exchange value in the IPsec SA output appears only after an IPsec SA rekey.

When the show commands display the configured ML-KEM algorithm and the tunnel status is READY, the quantum-safe hybrid key exchange is active.

Troubleshooting PQC hybrid key exchange

Use these debug commands to troubleshoot PQC hybrid key exchange issues on a Cisco IoT router.

Follow these steps to debug the PQC hybrid key exchange.

Procedure

Step 1 Use the **debug crypto ikev2** commands to debug IKEv2 negotiation, errors, packets, and internal events.

Example:

```
Router# debug crypto ikev2
Router# debug crypto ikev2 error
Router# debug crypto ikev2 packet
Router# debug crypto ikev2 internal
```

Step 2 Use the **debug crypto kmi** command to debug Key Management Infrastructure (KMI) events.

Example:

```
Router# debug crypto kmi
```

Step 3 Use the **debug crypto socket** command to debug crypto socket events.

Example:

```
Router# debug crypto socket
```

Step 4 Use the **debug crypto ipsec** commands to debug IPsec negotiation, errors, messages, and state changes.

Example:

```
Router# debug crypto ipsec
Router# debug crypto ipsec error
Router# debug crypto ipsec message
Router# debug crypto ipsec state
```

Step 5 Use the **debug tunnel protection** command to debug tunnel protection events.

Example:

```
Router# debug tunnel protection
```