KRYPTUS

APPLICATION NOTE · POST-QUANTUM CRYPTOGRAPHY

Post-Quantum Certification Authority in Active Directory Certificate Services

Issuing ML-DSA certificates on Windows Server 2025 with the signing key protected in the ASI-HSM AHX5 kNET

Document: Technical proof-of-concept guide
Version: 1.0 · **Date:** May 2026
Applies to: Windows Server 2025, AD CS, Kryptus kNET CNG provider (KSP), ASI-HSM AHX5 kNET
Classification: Public

Executive summary

The May 2026 update to Windows Server 2025 (KB5087539) enabled the post-quantum algorithm **ML-DSA** (FIPS 204) in **Active Directory Certificate Services (AD CS)**. Combined with the Kryptus CNG provider (KSP), it makes it possible to operate a **quantum-resistant Certification Authority (CA) whose signing key never leaves the HSM**.

This Application Note presents the context of the post-quantum transition and a **reproducible step-by-step** for: (1) updating Windows Server; (2) installing and provisioning the Kryptus CNG provider; (3) creating a CA in AD CS using ML-DSA with the key generated and protected in the **ASI-HSM AHX5 kNET**; and (4) **issuing certificates and certificate revocation lists (CRLs) signed with ML-DSA**.

1. Context

In August 2024 NIST finalized the first post-quantum standards: **FIPS 203 (ML-KEM)** for key encapsulation, **FIPS 204 (ML-DSA)** and **FIPS 205 (SLH-DSA)** for digital signatures. From there, industry adoption advanced quickly.

Microsoft brought post-quantum support in successive layers. In **September 2024**, ML-KEM and XMSS reached the SymCrypt cryptographic library. In **May 2025**, PQC appeared in the CNG APIs of Windows Insiders and Linux in early access. In **late 2025**, the ML-KEM and ML-DSA APIs reached general availability on Windows Server 2025 and Windows 11 24H2/25H2, already integrated into CNG and the certificate functions. In **May 2026**, update **KB5087539** (build 26100.32860) brought ML-DSA to AD CS, enabling certificate issuance by the PKI role.

On the Kryptus side, work with post-quantum algorithms began in **2020**; the milestone came in **2024**, when the **ASI-HSM AHX5 kNET** obtained **NIST CAVP** certification for its ML-DSA and ML-KEM implementation. The HSM generates and safeguards ML-DSA keys (and other PQC families) inside the secure hardware, and the Kryptus CNG provider (KSP) integrates it into Windows as a native *Key Storage Provider* that, with the AD CS update, now backs end-to-end post-quantum CAs.

1.1 About this proof of concept

This document is a **technical proof-of-concept (PoC) guide**. The step-by-step in the following chapters was validated in a lab environment, with example values (endpoints, credentials, server and Certification Authority names). Each figure corresponds to the actual output of that lab.

Because it is a PoC, this guide **does not replace a production deployment project**. Before applying the configuration described here to your PKI, adapt each step to your environment's security policies, network topology, capacity sizing, and compliance requirements. Validate the whole setup in a staging environment before migrating to production.

2. Prerequisites

- **Windows Server 2025** for the CA(s), with update **KB5087539 (2026-05)** or later (build 26100.32860+). For a test environment, download the evaluation image from the Microsoft Evaluation Center (ISO or VHD ready for Hyper-V).
- **AD CS role installed** on Windows Server 2025: *Server Manager → Manage → Add Roles and Features → Active Directory Certificate Services*, selecting the *Certification Authority* service. The CA *configuration* (choosing the kNET ML-DSA provider) is done in the step-by-step of §3.3.
- **Kryptus CNG provider (KSP)** for kNET (signed MSI installer).
- **ASI-HSM AHX5 kNET** reachable over the network, with firmware that supports ML-DSA, and a user/operator with permission to create and use keys.
- Administrative account on the CA server.
- **To validate the issued ML-DSA certificates** and verify their trust chain, you need a client machine with post-quantum cryptography support: **Windows 11 24H2/25H2 with update KB5067036 (Oct/2025) or later**. Clients without PQC support simply do not recognize ML-DSA certificates.

3. Step-by-step

3.1 Update Windows Server 2025

1. Install the cumulative update **KB5087539 (2026-05)**. There are three equivalent ways; choose the one that fits your environment:
   - **Windows Update (simplest):** in *Settings → Windows Update*, click *Check for updates*. The package is downloaded and installed automatically.
   - **Offline package (.msu):** download the file from the Microsoft Update Catalog and install it by **double-clicking** (opens the *Windows Update Standalone Installer*) or via the command line:

     ```
     PS> wusa.exe .\windows11.0-kb5087539-x64.msu /quiet /norestart
     ```

   - **Enterprise management:** in managed fleets, the update is distributed automatically by the corporate update servers, such as **WSUS** (*Windows Server Update Services*) or Microsoft Intune.

   Restart the server when prompted.

2. Confirm the build after the reboot:

   ```
   PS> $cv = Get-ItemProperty 'HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion'
   PS> "$($cv.CurrentBuild).$($cv.UBR)"
   26100.32860             # expected build (or higher)
   ```

3.2 Install and provision the Kryptus CNG provider (KSP)

3. Run the signed installer `knet-ksp-<version>.msi` and follow the wizard: accept the license agreement and finish with *Install → Finish*. The provider is registered in Windows as `kNET Key Storage Provider` (Figure 1).

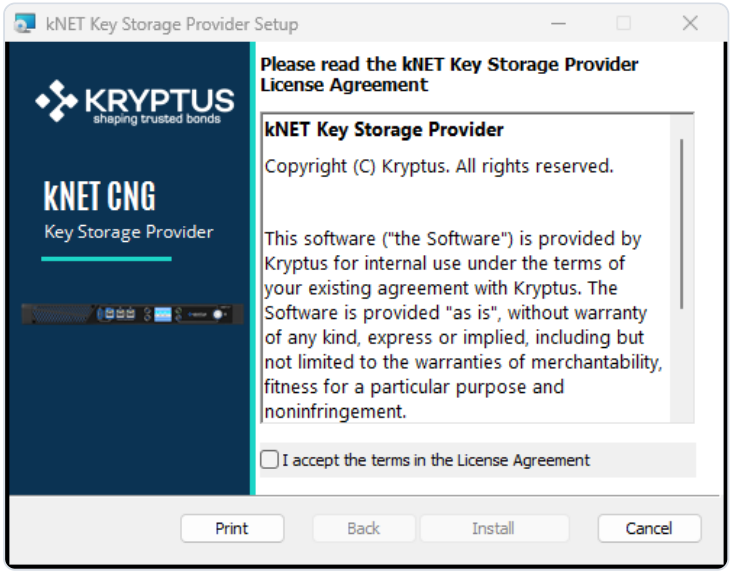

Figure 1. The Kryptus CNG provider installer.

4. Open the **KSP graphical tool**. On the *Connection* tab, enter one or more *HSM hosts* (one per line; multiple endpoints for cluster/high availability), the *Port*, the *Username*, and the operator password. Check *Verify TLS* and click **Fetch TLS Conf** so the tool fetches the HSM's TLS chain and certificate and fills in the corresponding fields. Use **Test Connection** to validate and click **Apply** to save (Figure 2).

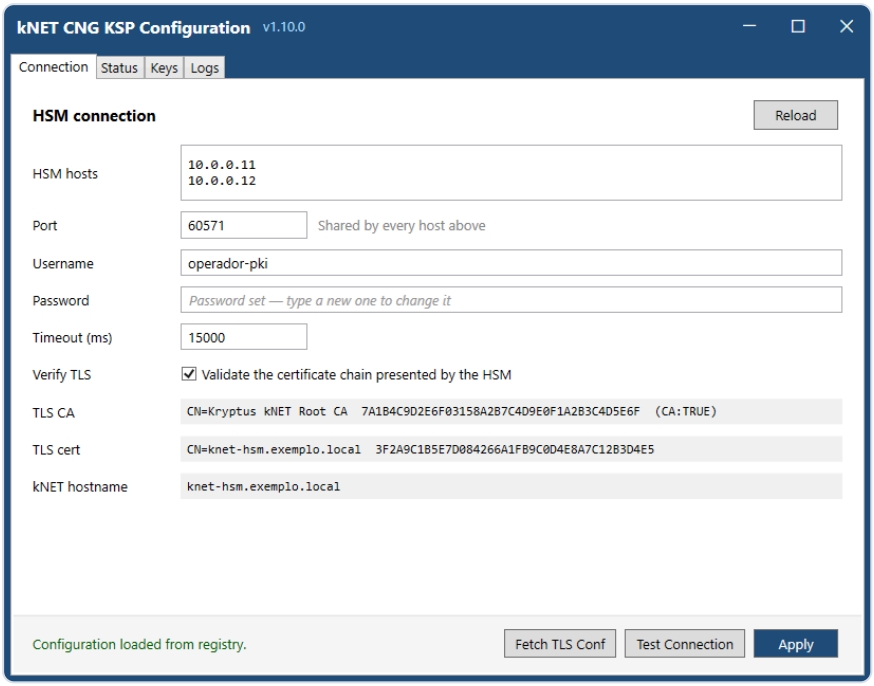

Figure 2. KSP graphical tool: connection to the HSM, with the TLS fields filled by *Fetch TLS Conf*.

From here the `kNET Key Storage Provider` is available to AD CS, including the `ML-DSA:44/65/87#kNET Key Storage Provider` sets.

3.3 Create the post-quantum CA in AD CS

There are two ways to do this, and the result is the same in both: the **graphical wizard** and the **command line**. Use whichever you prefer; in both, the ML-DSA key is generated and stays inside the HSM.

3.3.1 Via the graphical interface (GUI)

5. Install the *Active Directory Certificate Services* role (Server Manager → *Add Roles and Features*). Then, from the Server Manager notification, open the **AD CS configuration wizard** (*Configure Active Directory Certificate Services*). The first page, *Credentials*, confirms the administrative account that will run the configuration (Figure 3). Next, on the *Role Services* page, select **Certification Authority** (Figure 4).

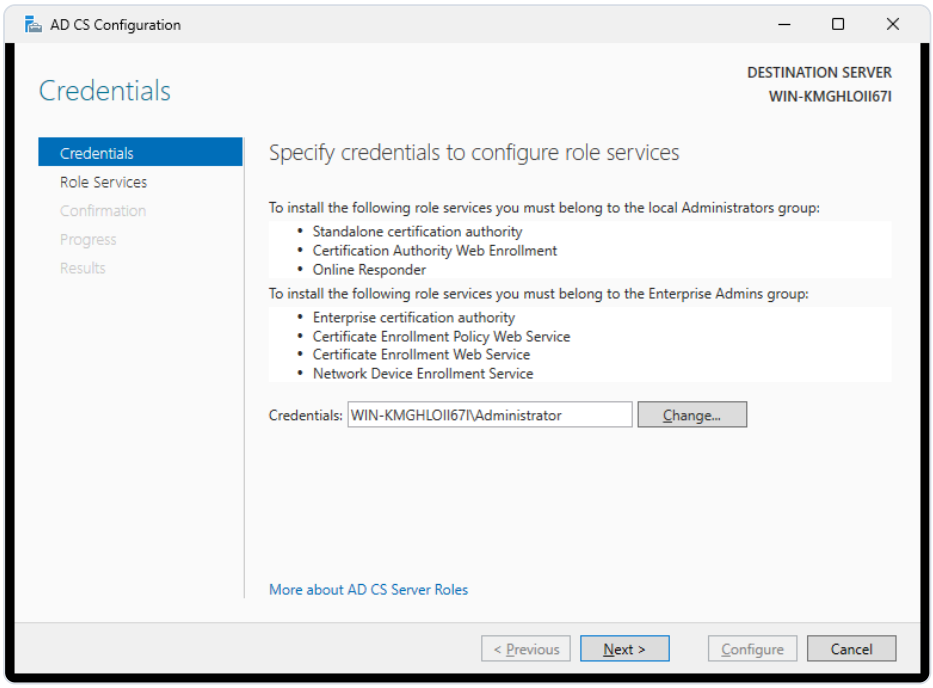

Figure 3. *Credentials* page: administrative account that runs the configuration wizard.

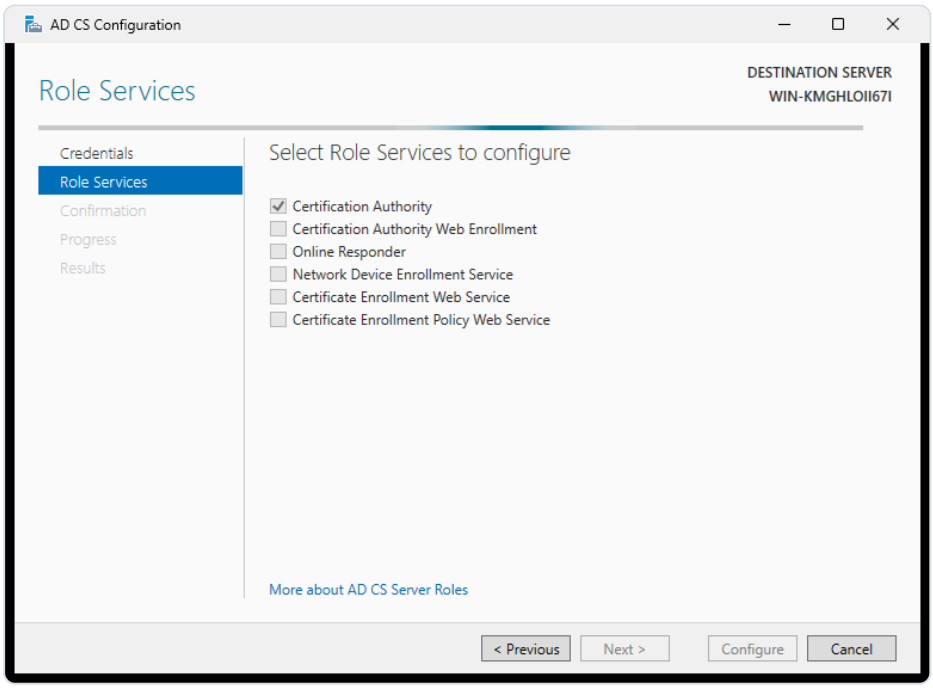

Figure 4. *Role Services* page with **Certification Authority** selected.

6 Next, choose the type and the key: **Standalone CA** (Figure 5), **Root CA** (Figure 6), and **Create a new private key** (Figure 7).

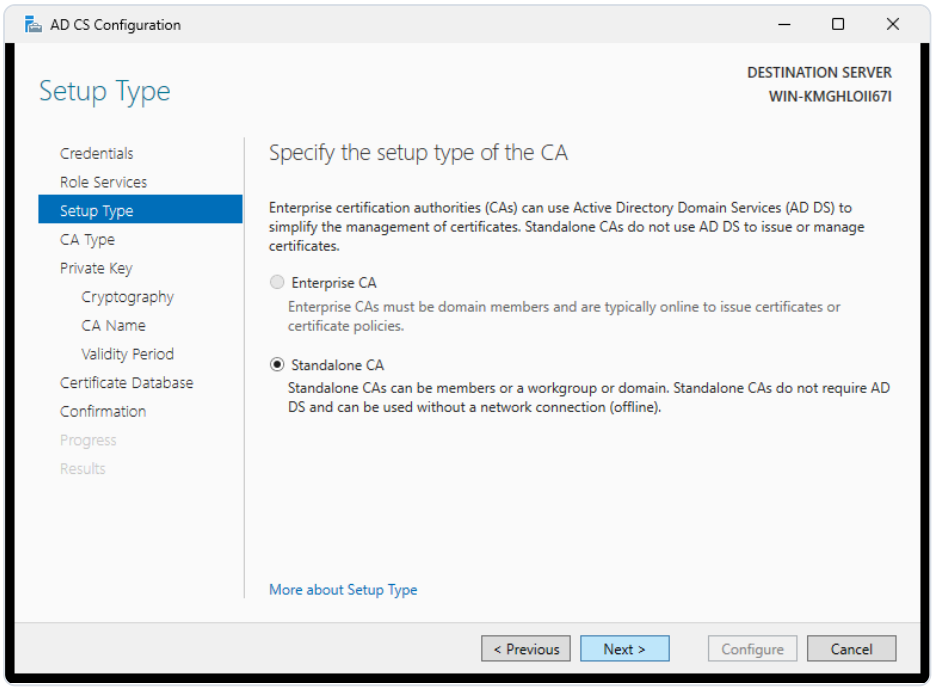

Figure 5. *Setup Type*: **Standalone CA**.

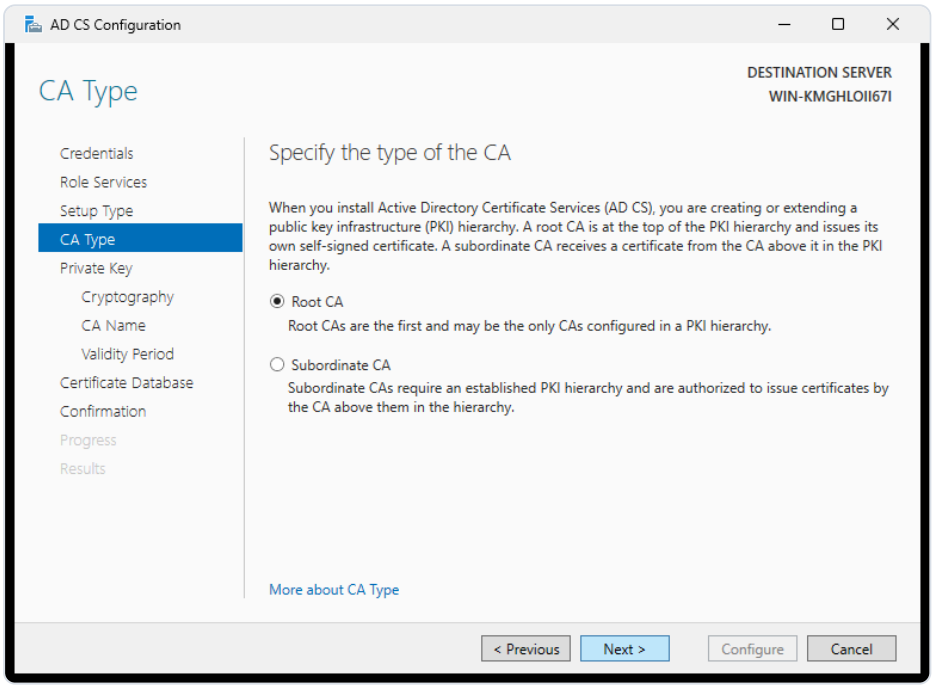

Figure 6. *CA Type*: **Root CA**.

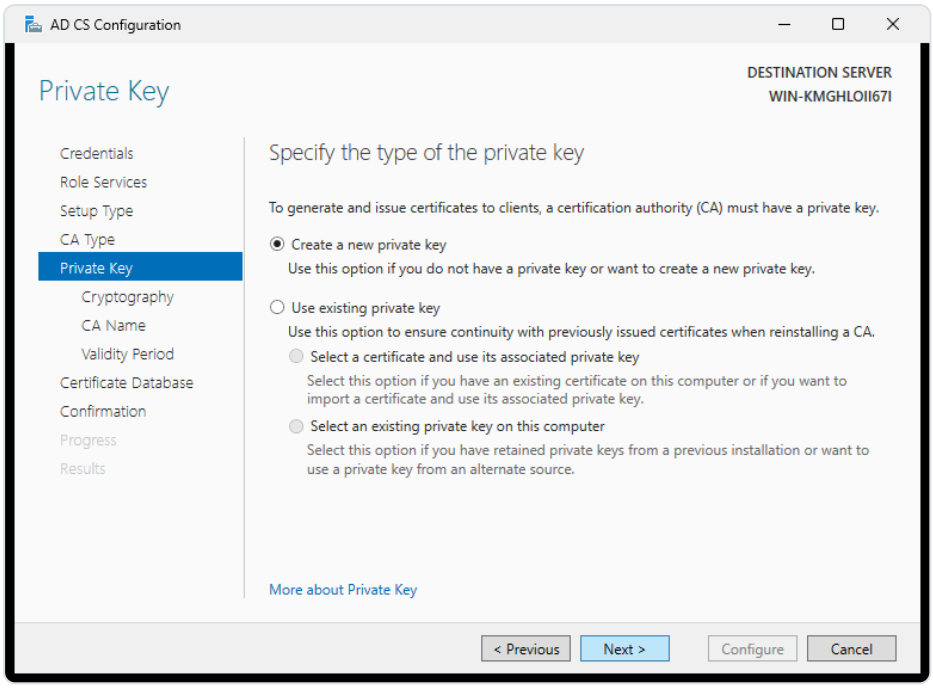

Figure 7. *Private Key*: **Create a new private key**.

7 On the *Cryptography for CA* page, open the provider list: the Kryptus KSP publishes the ML-DSA sets (in addition to RSA/ECDSA/Brainpool/DSA).

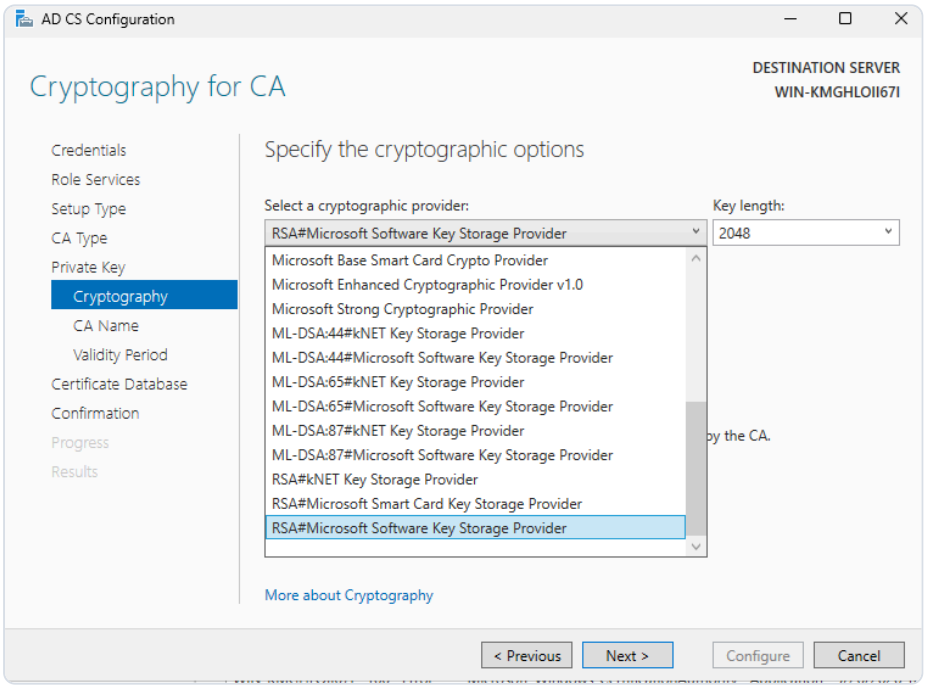

Figure 8. Cryptographic provider list: `ML-DSA:44/65/87#kNET Key Storage Provider` offered to the CA.

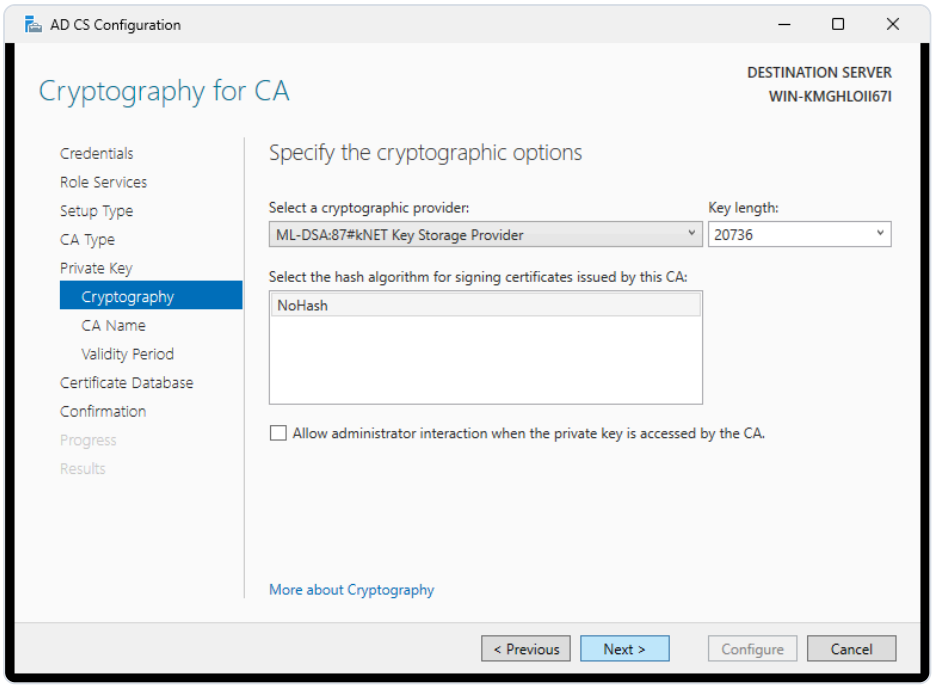

Figure 9. `ML-DSA:87#kNET Key Storage Provider` selected.

8 Set the CA name (Figure 10), the validity period of its certificate (Figure 11), and the location of the database and logs (Figure 12 — keep the default paths unless your policy requires another disk/volume).

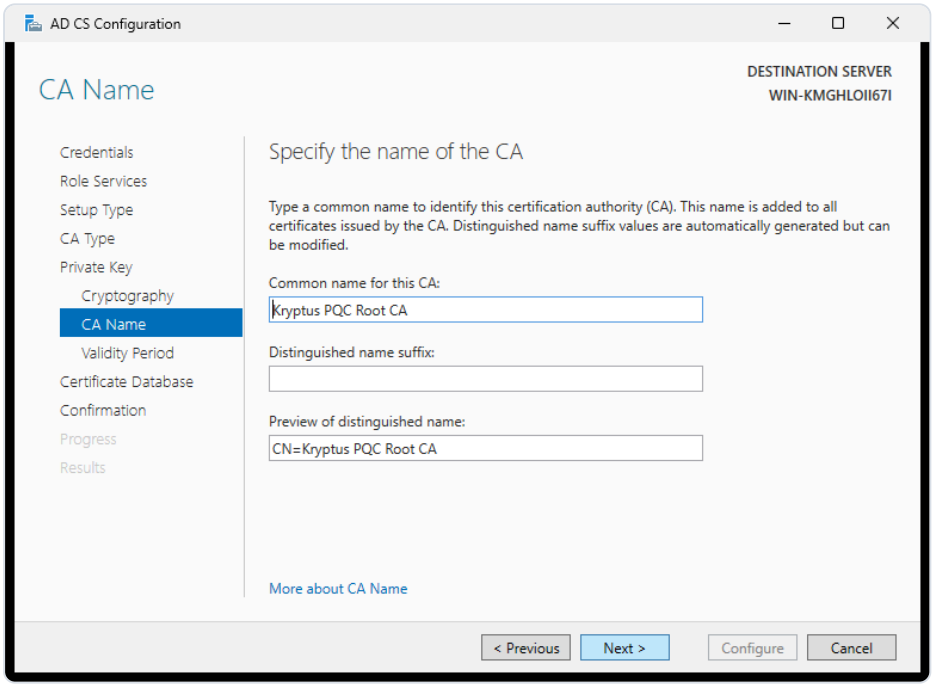

Figure 10. CA common name.

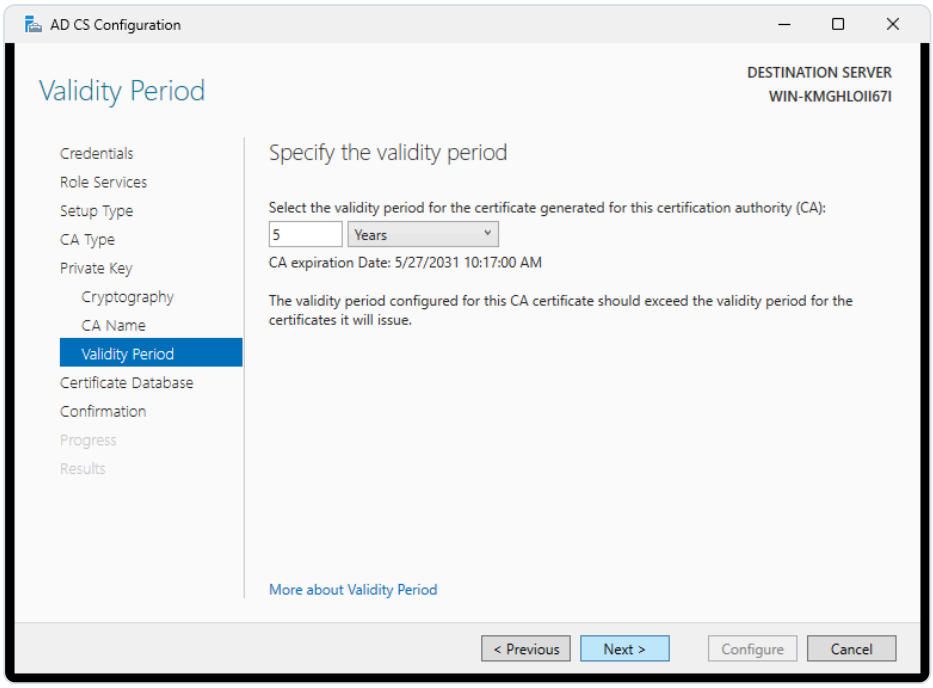

Figure 11. Validity period of the CA certificate.

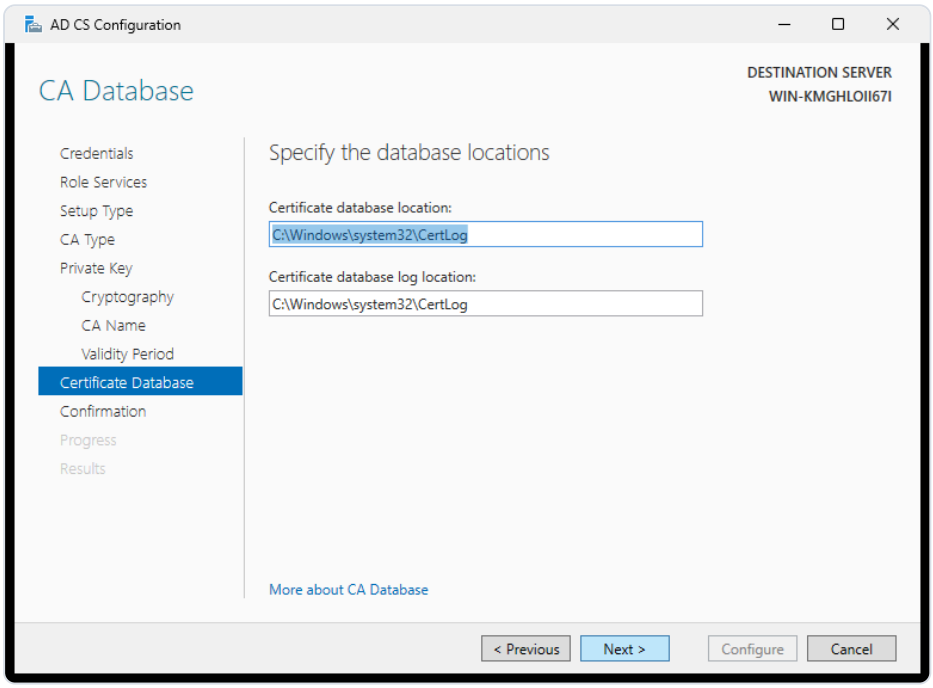

Figure 12. *Certificate Database* page: location of the CA database and log files.

9 Review the settings and click **Configure**: the ML-DSA key is generated in the HSM and the CA's self-signed certificate is produced with it (Figures 13 and 14).

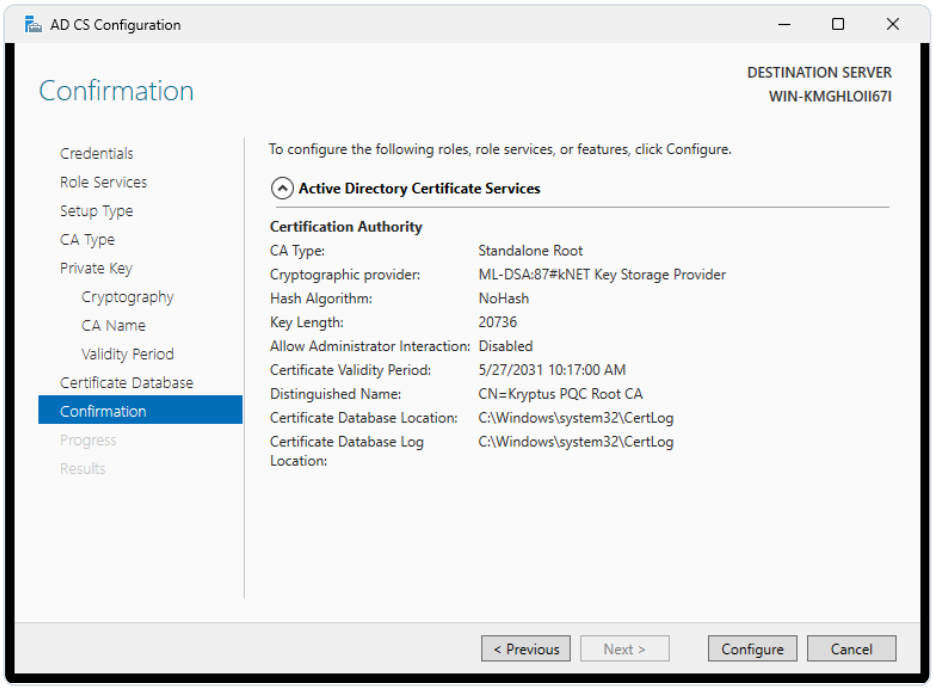

Figure 13. Review of the settings before creating the CA.

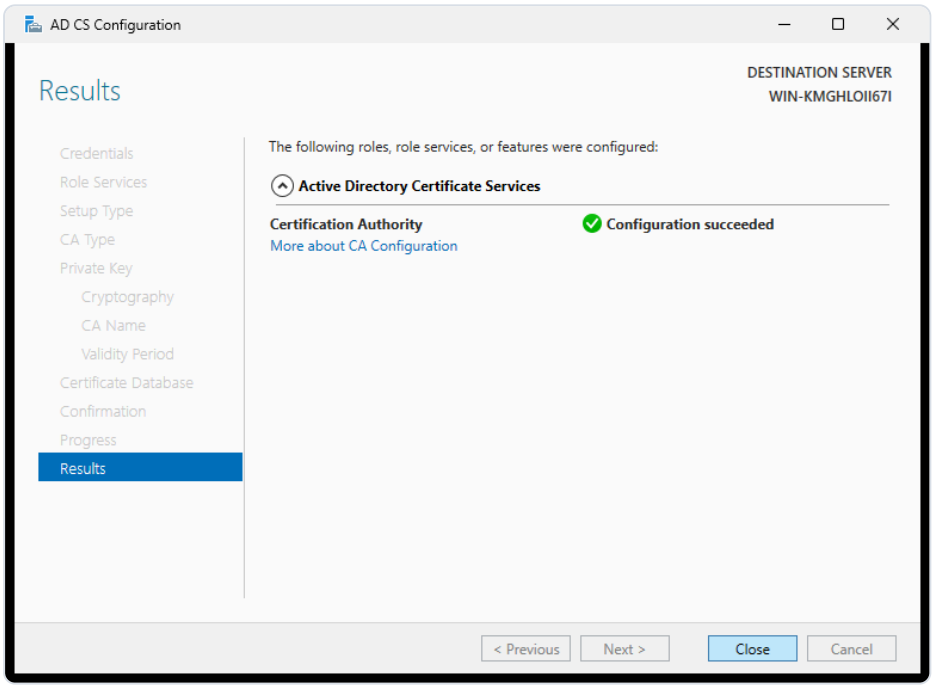

Figure 14. CA created successfully.

10 Open the Certification Authority console (`certsrv.msc` or *Server Manager → Tools → Certification Authority*) to confirm the CA is **running** (Figure 15). To inspect the certificate issued for the CA itself and the algorithm used, right-click the CA node → *Properties* → *View Certificate* → *Details tab*: the *Signature algorithm* and *Public key* fields show **ML-DSA-87** (Figure 16).

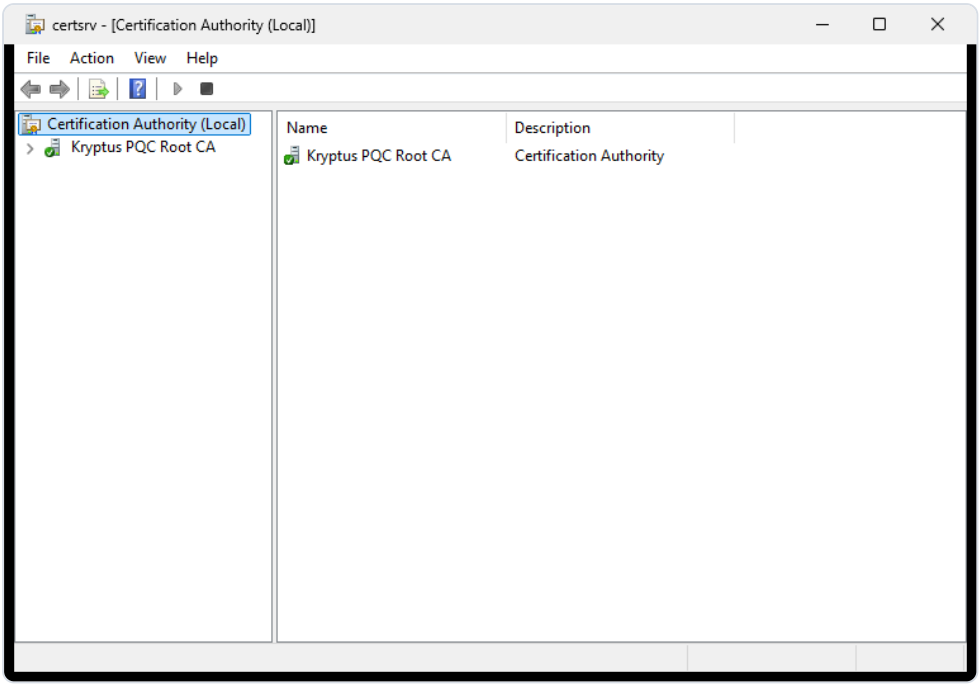

Figure 15. `certsrv.msc` : the *Kryptus PQC Root CA* running.

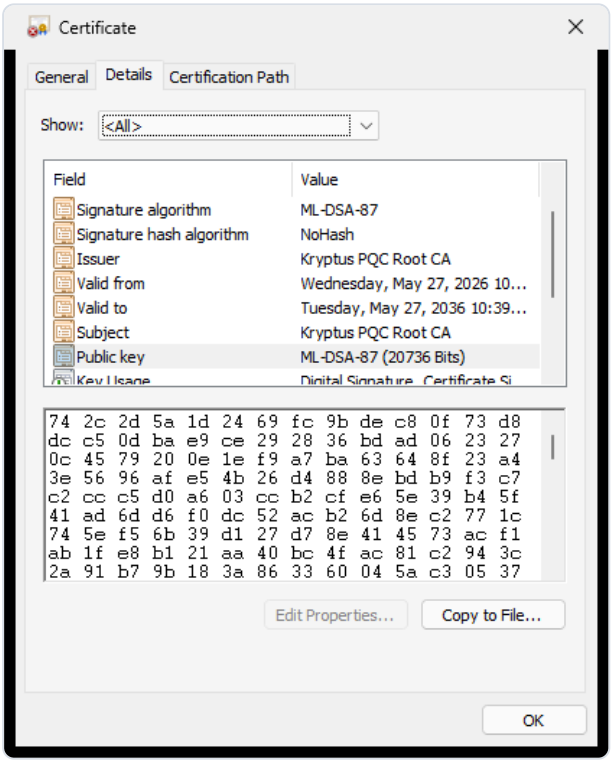

Figure 16. CA certificate details: *Signature algorithm* and *Public key* as **ML-DSA-87** (20736 bits).

11 Verify that the CA's signing key resides in the HSM:

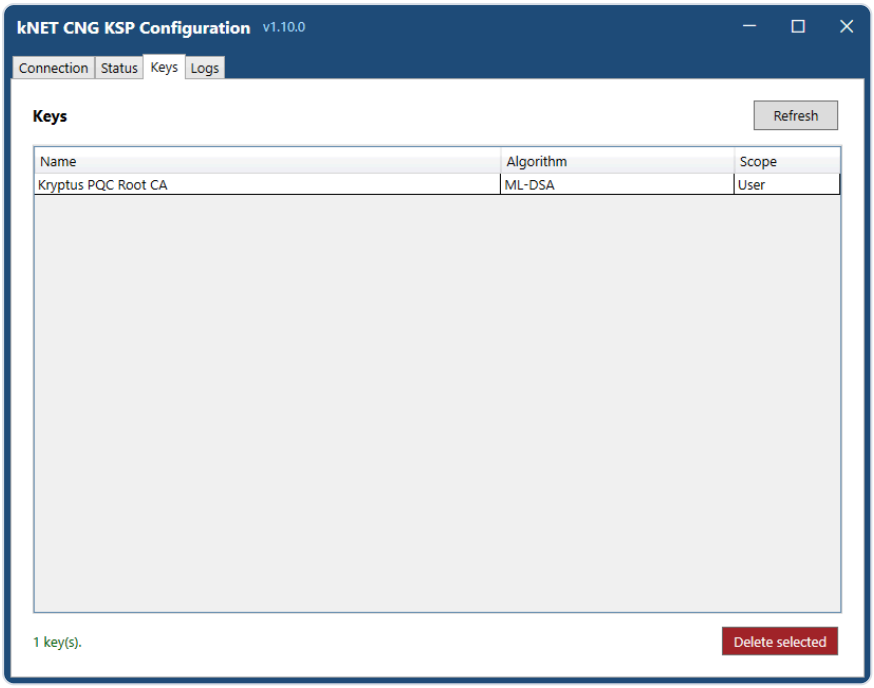

Figure 17. The CA key listed in the HSM by the KSP tool.

3.3.2 Via the command line (CLI)

Creating the CA can also be done from the command line. The blocks below reproduce the real output of the commands run in the lab (abbreviated in the longer passages); the server name was anonymized as `<server-name>`.

5 Create the ML-DSA-87 CA.

```
PS> Install-AdcsCertificationAuthority `
      -CAType             StandaloneRootCA `
      -CACommonName       "Kryptus PQC Root CA" `
      -CryptoProviderName "ML-DSA:87#kNET Key Storage Provider" `
      -HashAlgorithmName  NoHash -KeyLength 20736 `
      -ValidityPeriod     Years -ValidityPeriodUnits 10 -Force

ErrorId ErrorString
------- -----------
      0          # ErrorId 0: installation completed successfully
```

6 Confirm the CA and the ML-DSA-87 algorithm.

```
PS> certutil -CAInfo
CA name: Kryptus PQC Root CA
CA type: 3 -- Stand-alone Root CA
    ENUM_STANDALONE_ROOTCA -- 3
CA cert[0]: 3 -- Valid
CRL[0]: 3 -- Valid
DNS Name: <server-name>
CertUtil: -CAInfo command completed successfully.

PS> certutil -dump ca.cer          # ca.cer = certificate exported from the CA
Signature Algorithm:
    Algorithm ObjectId: 2.16.840.1.101.3.4.3.19 ML-DSA-87
Public Key Algorithm:
    Algorithm ObjectId: 2.16.840.1.101.3.4.3.19 ML-DSA-87
Public Key Length: 20736 bits
```

3.4 Issue, revoke, and publish certificates (CRL)

With the CA ready, the certificate lifecycle (issuance, revocation, and CRL publication) can be carried out via the graphical interface or the command line. In both paths, the leaf key is generated and stays inside the HSM, and all signatures (certificate and CRL) use the CA's ML-DSA key in the hardware.

3.4.1 Via the graphical interface (GUI)

The full flow via the GUI: generate the request with the key in the HSM (*Certificates* snap-in, steps 1 to 6) and operate the lifecycle in the CA console (`certsrv.msc` , steps 7 to 11).

1 In the *Certificates* snap-in (`certmgr.msc` for the user or `certlm.msc` for the machine), right-click *Personal* → *All Tasks* → *Advanced Operations* → *Create Custom Request* (Figure 18).

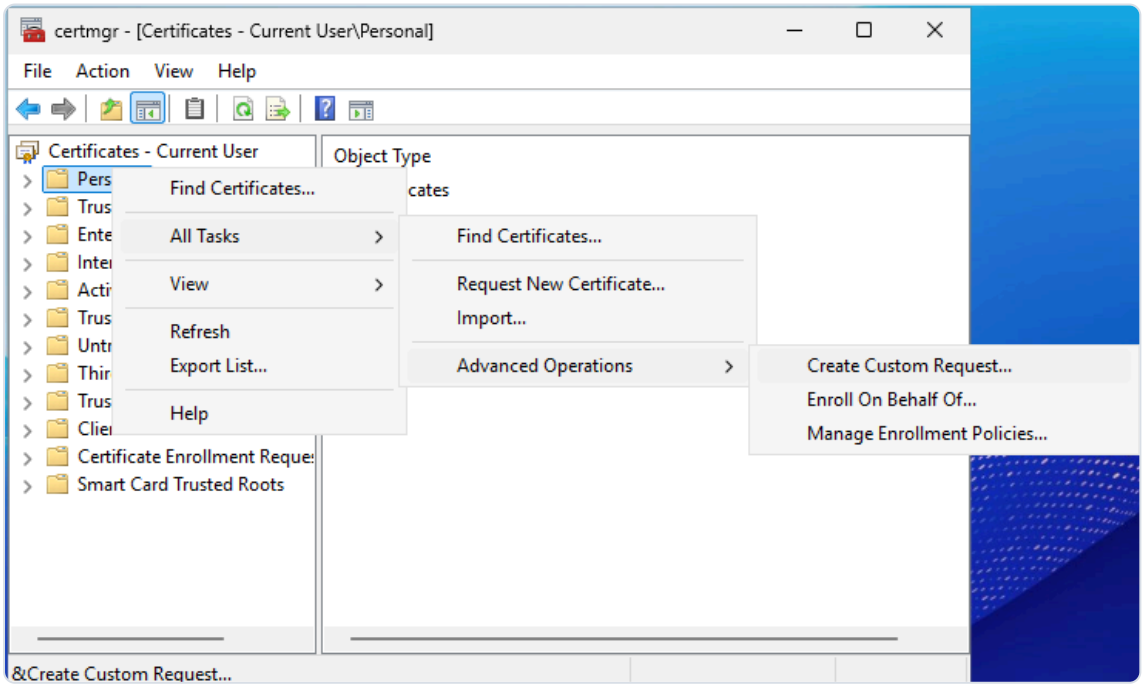

Figure 18. Starting the request in *Personal* → *All Tasks* → *Advanced Operations* → *Create Custom Request*.

2 In *Select Certificate Enrollment Policy*, choose **Proceed without enrollment policy** (custom request, no AD policy) (Figure 19).

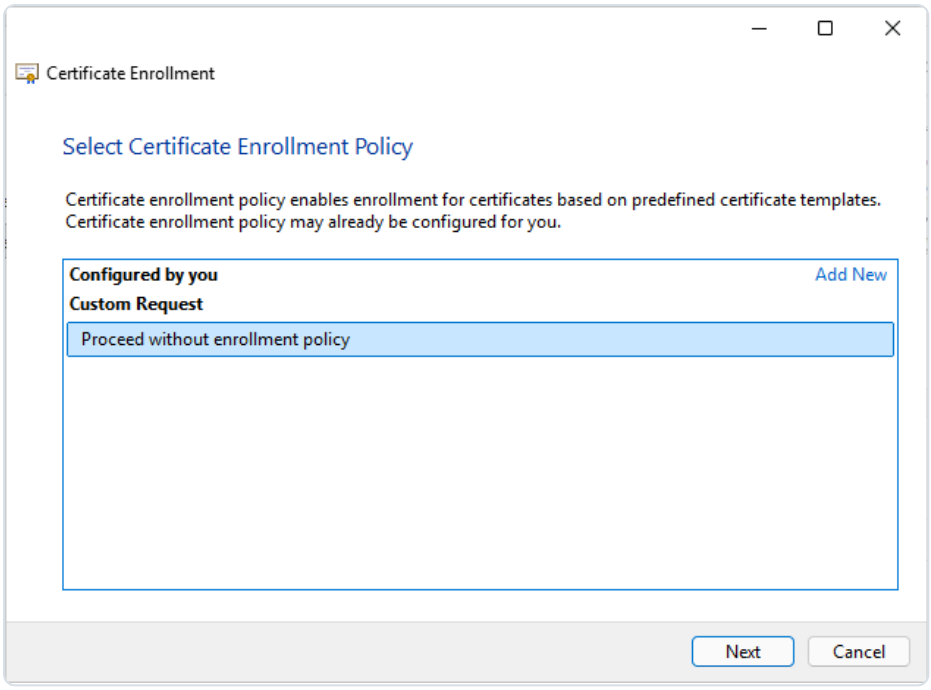

Figure 19. *Custom Request*: **Proceed without enrollment policy**.

3 In *Custom request*, select the template **"(No template) CNG key"** and the **PKCS #10** format (Figure 20). The KSP is a CNG provider, so the option must be *CNG key*, not *Legacy key*.

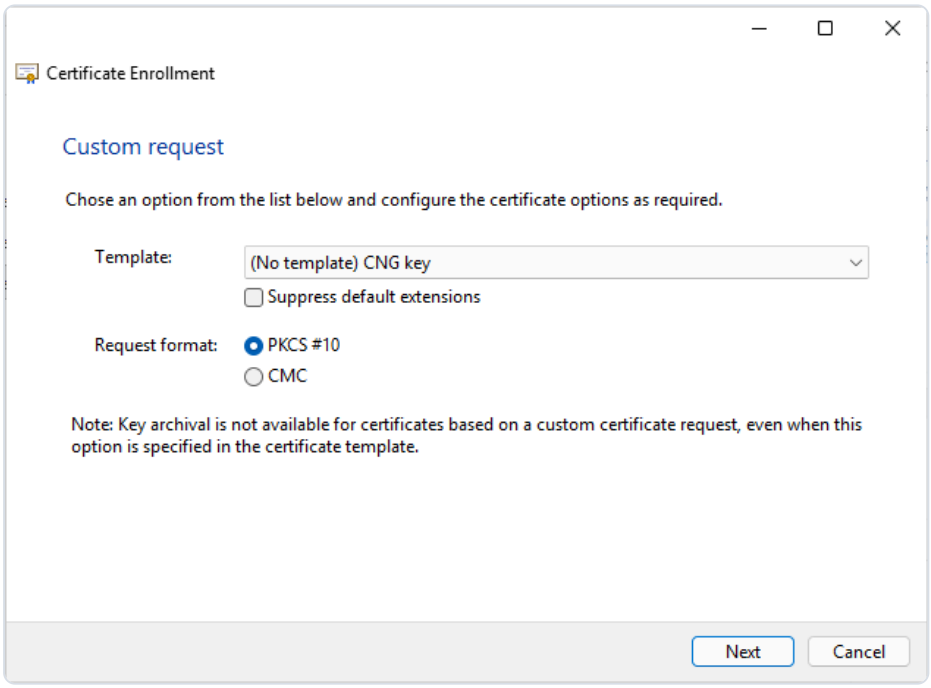

Figure 20. Template **(No template) CNG key** and **PKCS #10** format.

4. The *Certificate Information* screen summarizes the request; click *Details* → *Properties* to configure it (Figure 21). On the *Subject* tab, enter the subject (e.g., `CN=...`) (Figure 22); on the *Extensions* tab, adjust the *Key usage* (e.g., *Digital signature*) (Figure 23).

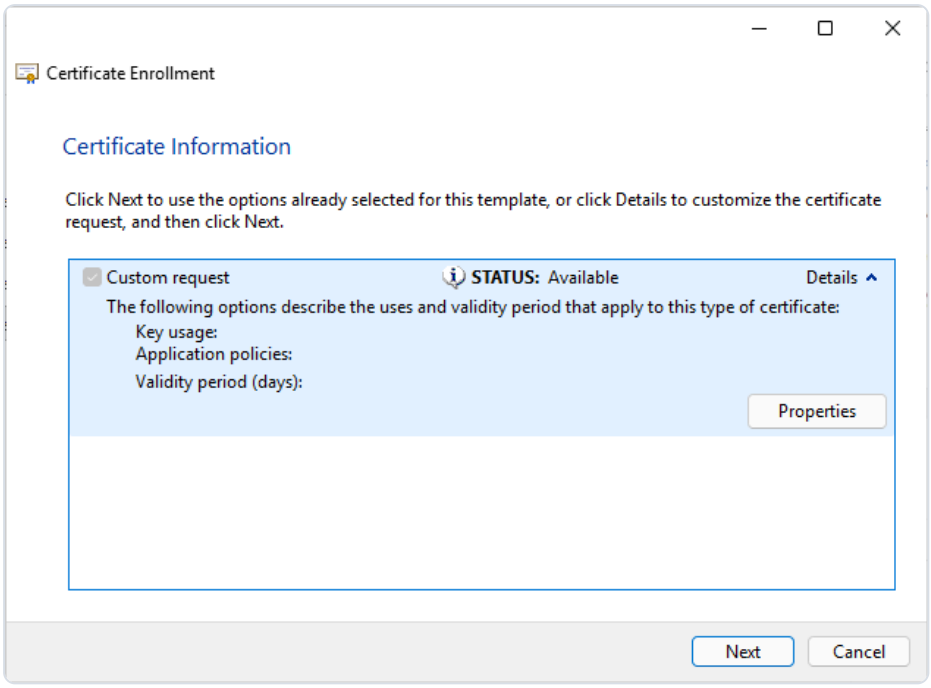

Figure 21. *Certificate Information* before configuring: click *Details* → *Properties*.

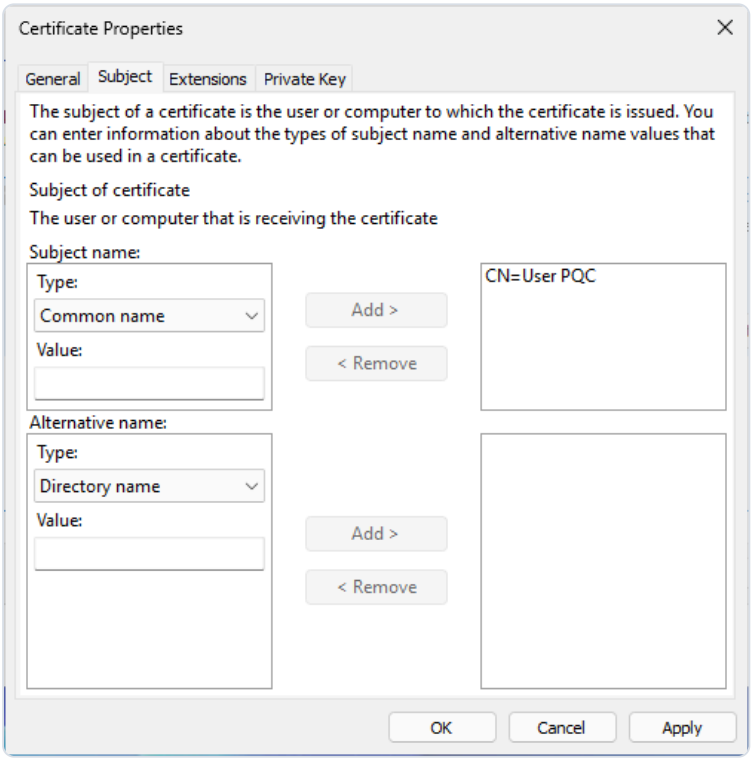

Figure 22. *Subject* tab: the certificate subject name.

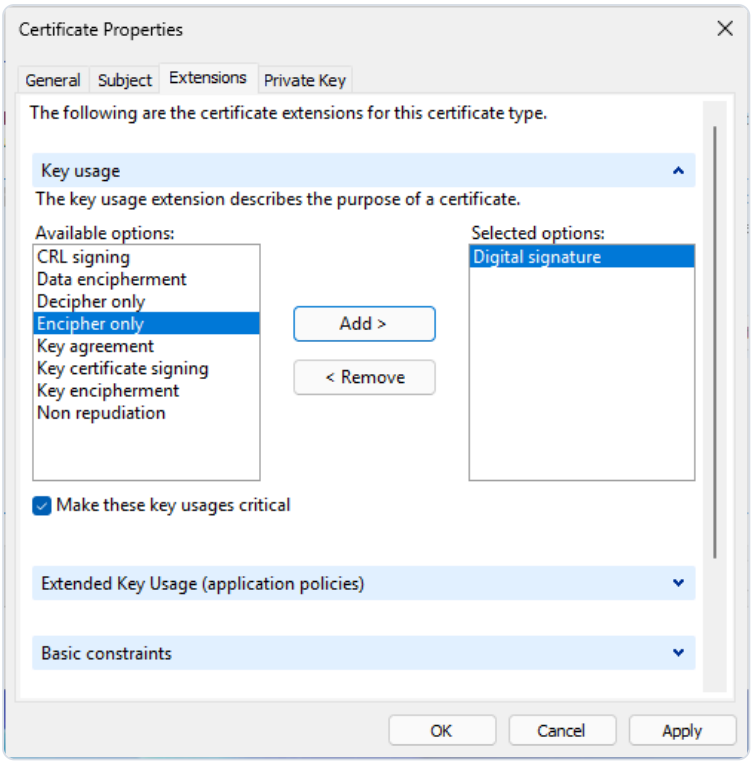

Figure 23. *Extensions* tab: the certificate's *Key usage* (e.g., *Digital signature*).

5 On the *Private Key* → *Cryptographic Service Provider* tab, select the desired **kNET Key Storage Provider** set, e.g., `ML-DSA:65, kNET Key Storage Provider` (Figure 24). This choice is what makes the key pair **be born inside the HSM**.

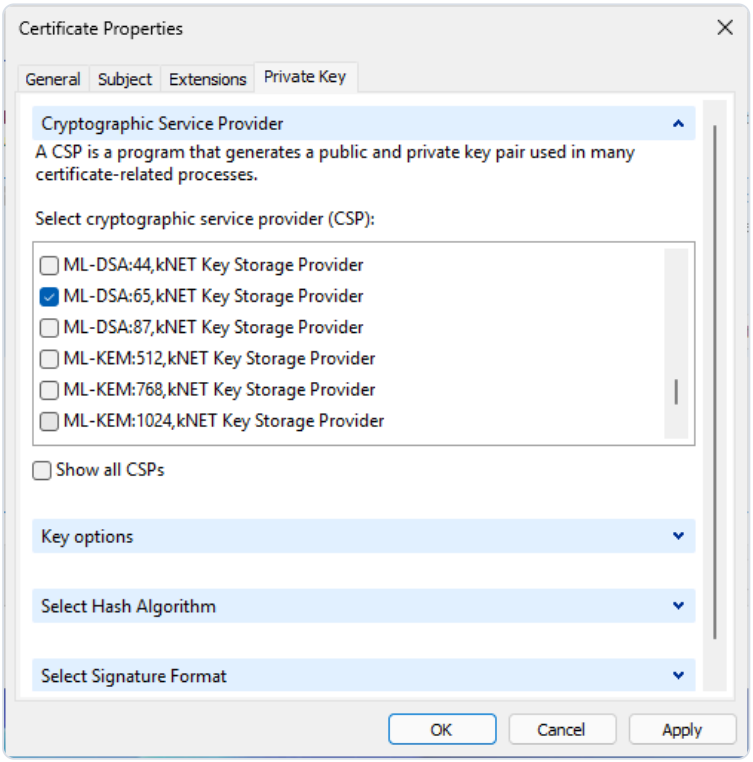

Figure 24. Provider `ML-DSA:65, kNET Key Storage Provider` selected: the key is generated in the HSM.

6 Back on the *Certificate Information* screen, now with the options applied (Figure 25), finish by saving the request (`.req`) in **Base 64** (Figure 26). The private key is already in the HSM; the file carries only the request (CSR).

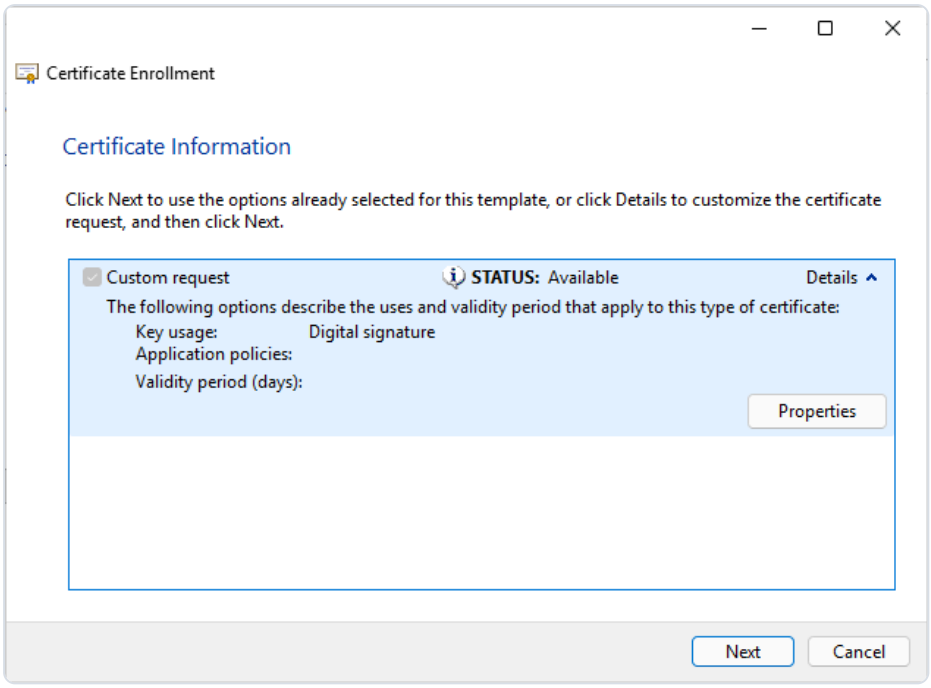

Figure 25. *Certificate Information* after the request is configured.

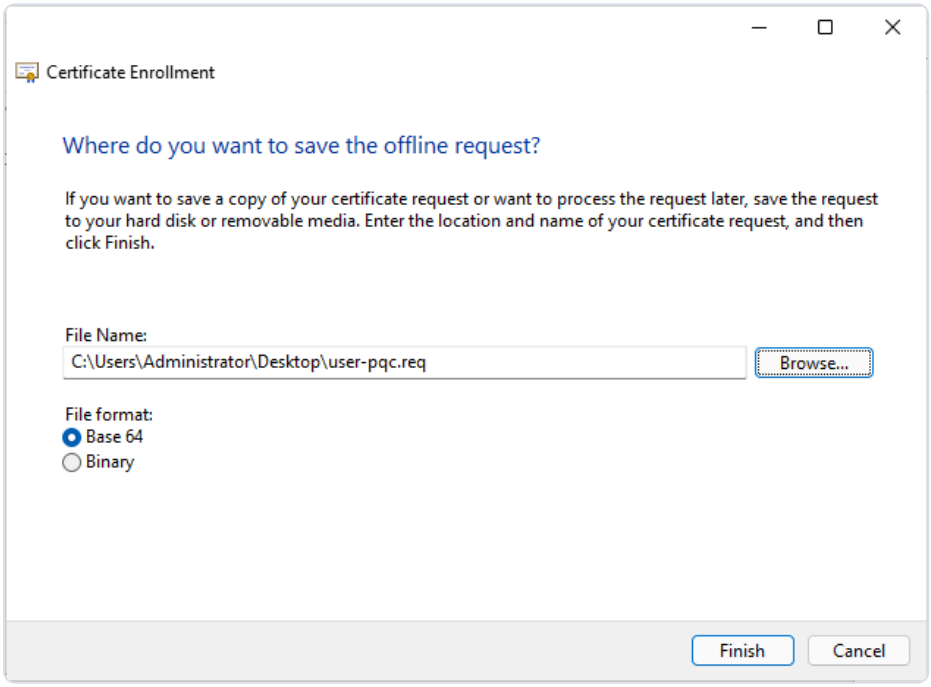

Figure 26. Request saved in **Base 64** (`.req`).

7 In the CA console (`certsrv.msc`), right-click the CA node → *All Tasks* → *Submit new request* and select the `.req` (Figure 27). On a *standalone* CA, the request comes in as pending.

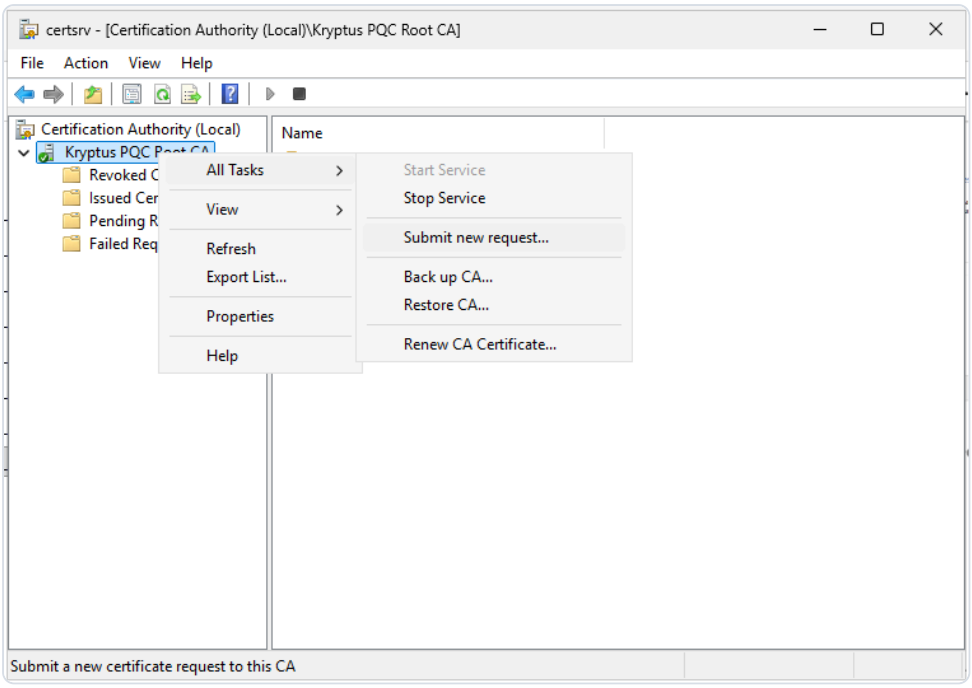

Figure 27. *All Tasks → Submit new request*: sending the CSR to the CA.

8 In *Pending Requests*, right-click the request → *All Tasks* → *Issue* to issue it (Figure 28). The certificate moves to *Issued Certificates*, from where it can be exported and inspected: the issued leaf has an **ML-DSA-65** public key and was signed by the CA with **ML-DSA-87** (Figure 29).

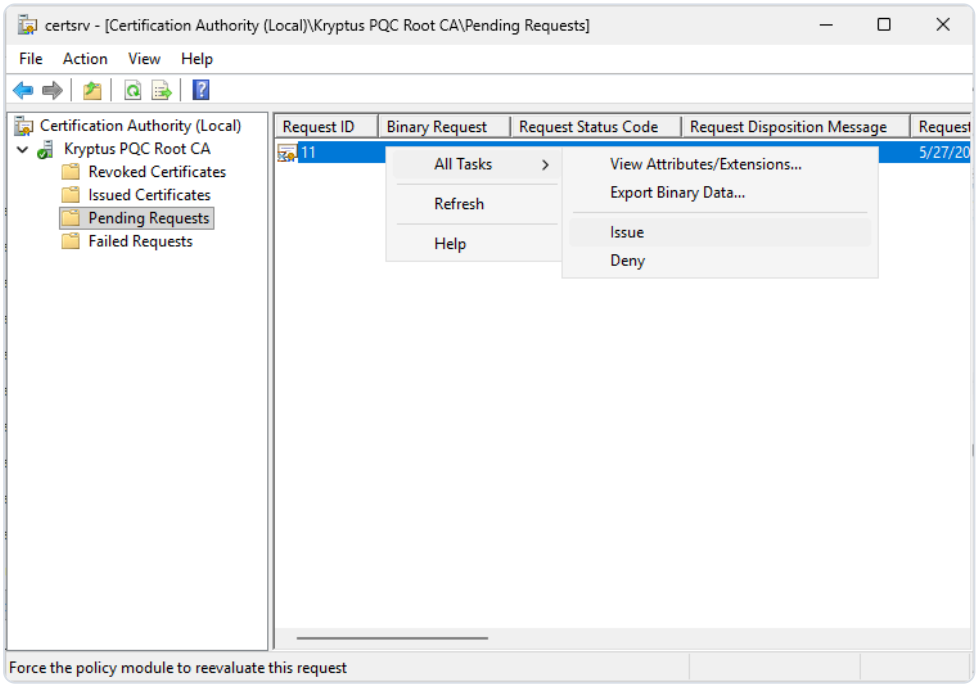

Figure 28. Approving the pending request in *All Tasks → Issue*.

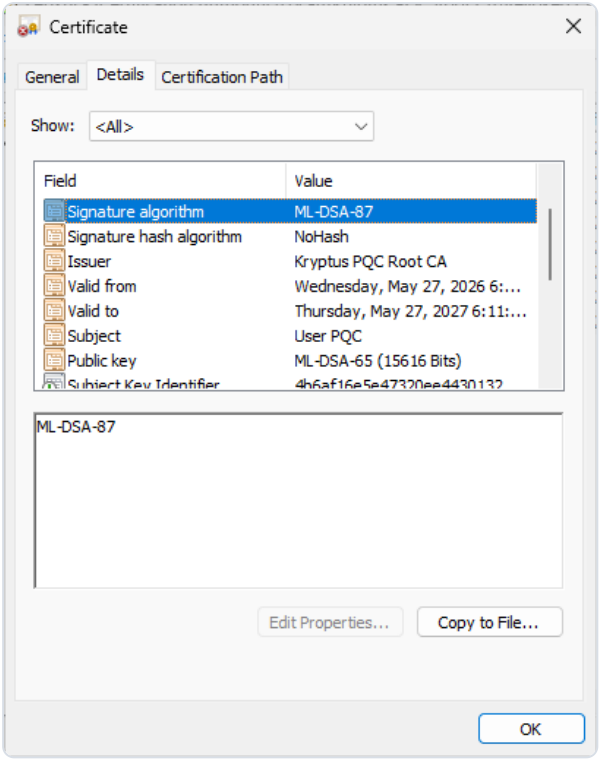

Figure 29. Issued certificate: *Public key* **ML-DSA-65** and *Signature algorithm* **ML-DSA-87**.

9 To **revoke**, in *Issued Certificates*, right-click the certificate → *All Tasks* → *Revoke Certificate* (Figure 30); choose the reason and confirm (Figure 31).

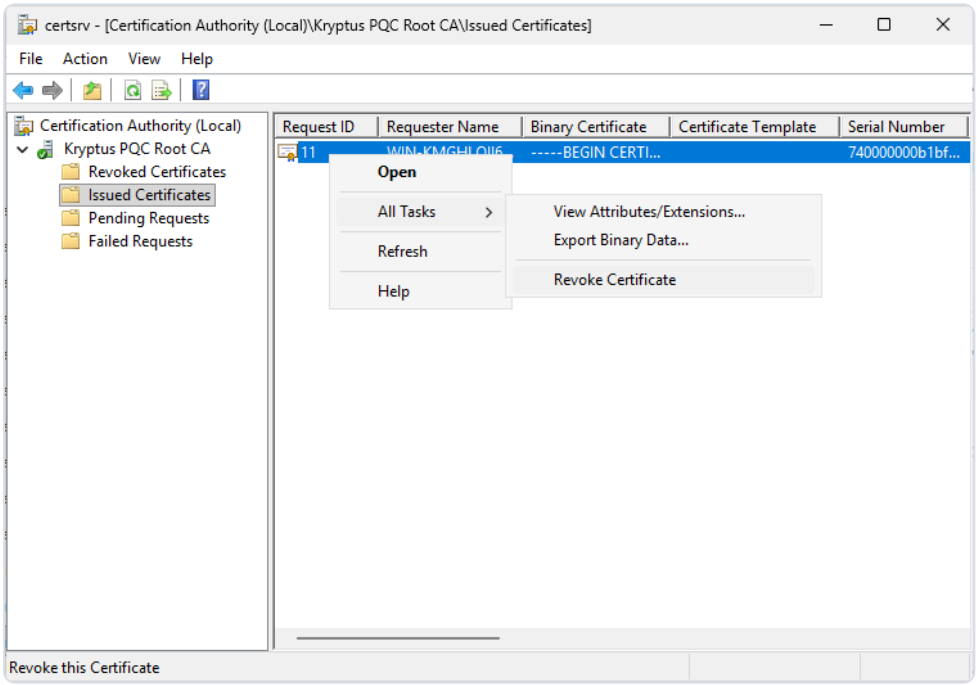

Figure 30. *All Tasks → Revoke Certificate.*

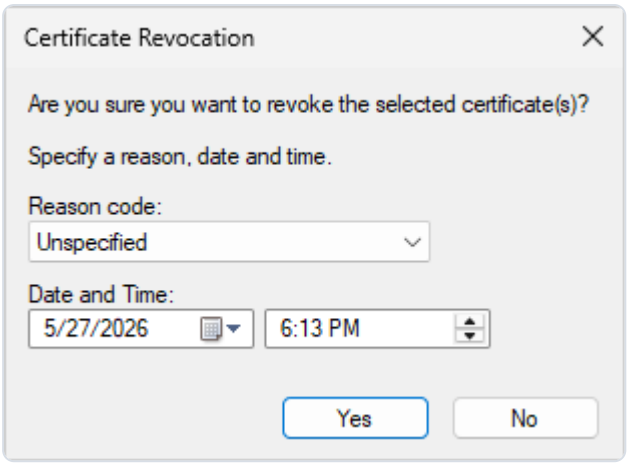

Figure 31. Revocation dialog: *Reason code* and date/time.

10 To **publish the CRL**, in *Revoked Certificates*, right-click → *All Tasks* → *Publish* (Figure 32) and choose **New CRL** (Figure 33).

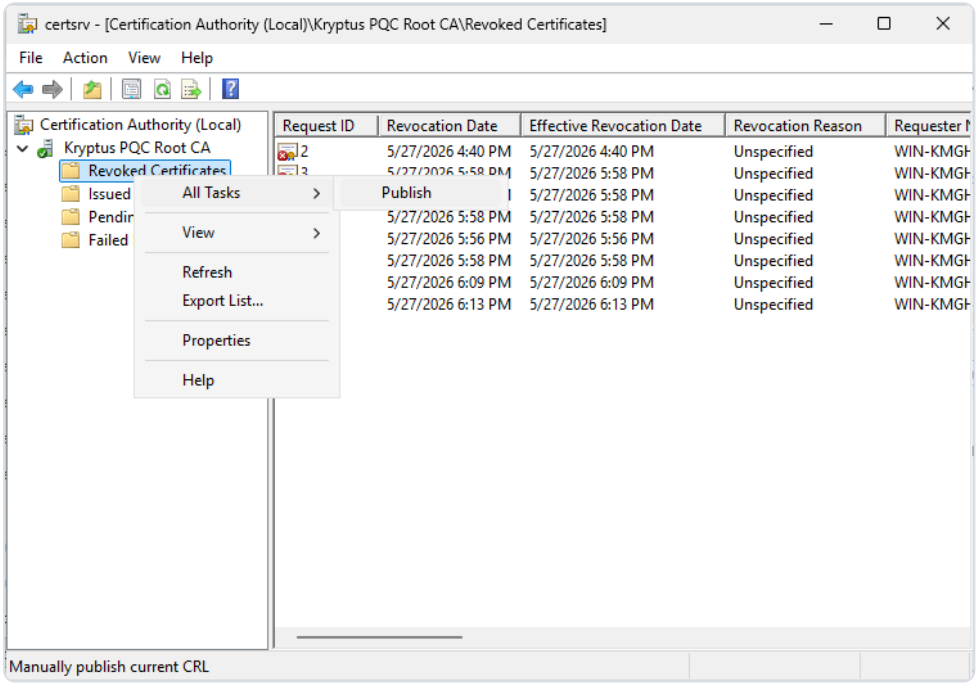

Figure 32. *All Tasks → Publish*: generating a new CRL.

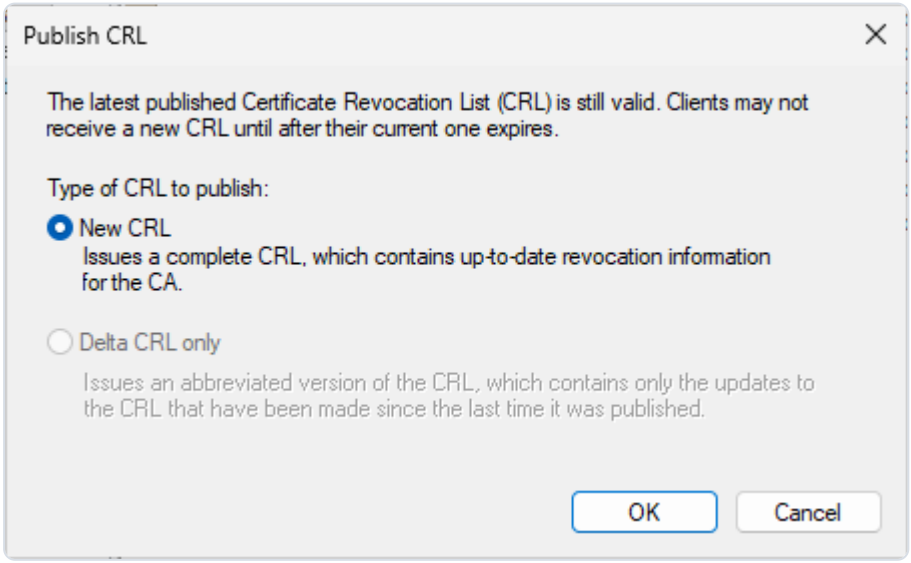

Figure 33. *Publish CRL*: **New CRL**.

11 The CRL is also signed with the CA's ML-DSA key. In *Revoked Certificates → Properties → View CRLs → View CRL*, the *Signature algorithm* field shows **ML-DSA-87** (Figure 34).

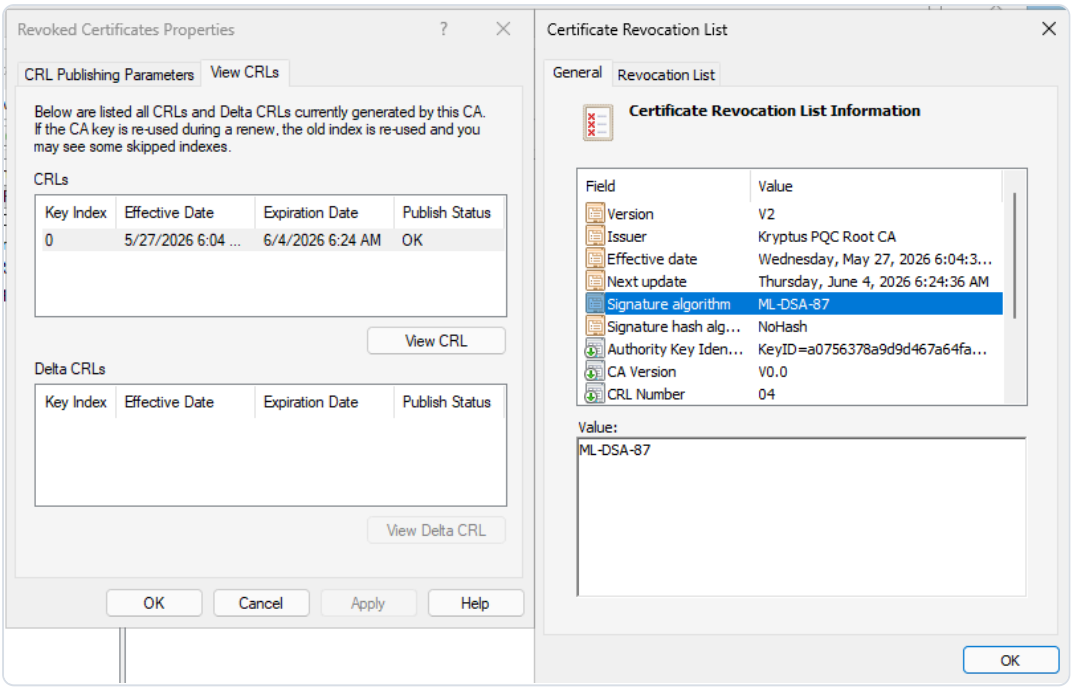

Figure 34. CRL details: *Signature algorithm* as **ML-DSA-87**.

3.4.2 Via the command line (CLI)

The same operations via the command line (real output, abbreviated in the longer passages; server anonymized as `<server-name>`):

1 Issue a certificate (ML-DSA-65 leaf).

```
# leaf.inf: request for a post-quantum leaf in the HSM
[NewRequest]
Subject      = "CN=pqc-server.example.local"
RequestType  = PKCS10
ProviderName = "kNET Key Storage Provider"
KeyAlgorithm = ML-DSA-65
MachineKeySet = TRUE
KeyUsage     = 0x80

PS> certreq -q -new -f leaf.inf leaf.req
CertReq: Request Created
PS> certreq -q -config "<server-name>\Kryptus PQC Root CA" -submit leaf.req leaf.cer
RequestId: 2
RequestId: "2"
Certificate request is pending: Taken Under Submission (0)
PS> certutil -resubmit 2                # the CA operator approves
Certificate issued.
CertUtil: -resubmit command completed successfully.
PS> certreq -q -config "<server-name>\Kryptus PQC Root CA" -retrieve 2 leaf.cer
RequestId: 2
RequestId: "2"
Certificate retrieved(Issued) Issued  Resubmitted by <server-name>\Administrator
```

The issued certificate forms a **100% post-quantum chain**: ML-DSA-65 leaf key, signed by the CA with ML-DSA-87.

```
PS> certutil -dump leaf.cer
Serial Number: 740000000234fa2471b0af215a000000000002
Signature Algorithm:
    Algorithm ObjectId: 2.16.840.1.101.3.4.3.19 ML-DSA-87
Issuer:
    CN=Kryptus PQC Root CA
Subject:
    CN=pqc-server.example.local
Public Key Algorithm:
    Algorithm ObjectId: 2.16.840.1.101.3.4.3.18 ML-DSA-65
Public Key Length: 15616 bits
```

2. Revoke and publish the CRL.

```
PS> certutil -revoke 740000000234fa2471b0af215a000000000002
Revoking "740000000234fa2471b0af215a000000000002" -- Reason: Unspecified
CertUtil: -revoke command completed successfully.
PS> certutil -CRL
CertUtil: -CRL command completed successfully.  # CRL signed with ML-DSA-87
```

3. Verify the chain and the revocation.

```
PS> certutil -verify leaf.cer
ChainContext.dwErrorStatus = CERT_TRUST_IS_REVOKED (0x4)

CertContext[0][0]: dwInfoStatus=102 dwErrorStatus=4
  Issuer: CN=Kryptus PQC Root CA
  Subject: CN=pqc-server.example.local
  Serial: 740000000234fa2471b0af215a000000000002
  Element.dwErrorStatus = CERT_TRUST_IS_REVOKED (0x4)
    CRL 03:
    Issuer: CN=Kryptus PQC Root CA

CertContext[0][1]: dwInfoStatus=10c dwErrorStatus=0
  Subject: CN=Kryptus PQC Root CA   # self-signed root ML-DSA-87
  Element.dwInfoStatus = CERT_TRUST_IS_SELF_SIGNED (0x8)

The certificate is revoked. 0x80092010 (-2146885616 CRYPT_E_REVOKED)
Leaf certificate is REVOKED (Reason=0)
CertUtil: -verify command completed successfully.
```

4. Final remarks

Cryptographically relevant quantum computers threaten the classical asymmetric algorithms (RSA, ECDSA, Diffie-Hellman). The risk is immediate even before such machines exist, because of the "harvest now, decrypt later" attack: data and signatures captured today can be broken in the future. That is why migration must start now.

This shift is government-coordinated. In the United States, NIST published the standards (FIPS 203/204/205) and the NSA, through the **CNSA 2.0** suite, requires post-quantum cryptography for National Security Systems, with a preferred transition by 2030-2033 and a full one by **2035**; the White House (NSM-10) and OMB direct federal agencies to migrate on the same horizon. In Europe, the European Commission recommended, in 2024, that member states adopt a **coordinated roadmap** for the transition to PQC, favoring hybrid schemes and targeting around 2030 for high-risk systems and 2035 more broadly, supported by national guidance such as Germany's BSI and France's ANSSI. The message converges: plan and begin the migration now.

The post-quantum transition has moved out of the theoretical realm and reached the Windows enterprise PKI. With KB5087539 and the Kryptus CNG provider, it is possible, **today**, to operate an ML-DSA Certification Authority with the signing key protected in the **ASI-HSM AHX5 kNET**, combining quantum resistance and hardware security. We recommend starting with parallel-hierarchy pilots and mapping the points in your PKI that will need post-quantum signatures.

References

1. Microsoft's quantum-resistant cryptography is here (SymCrypt, 2024)
2. Post-Quantum Cryptography Comes to Windows Insiders and Linux (2025)
3. Post-Quantum Cryptography APIs Now Generally Available on Microsoft Platforms
4. What is ML-DSA support in AD CS? (Microsoft Learn)
5. Configure a certification authority to use ML-DSA (Microsoft Learn)
6. Configure certificate templates for ML-DSA (Microsoft Learn)
7. Update KB5087539 (2026-05-12, build 26100.32860)
8. NIST FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM), August 2024
9. NIST FIPS 204: Module-Lattice-Based Digital Signature Standard (ML-DSA), August 2024
10. NIST FIPS 205: Stateless Hash-Based Digital Signature Standard (SLH-DSA), August 2024
11. NIST CAVP: cryptographic implementation validation of the ASI-HSM AHX5 kNET (Kryptus)
12. NSA: Commercial National Security Algorithm Suite 2.0 (CNSA 2.0), the U.S. National Security Systems PQC transition
13. European Commission: Recommendation (EU) 2024/1101 on a Coordinated Implementation Roadmap for the transition to Post-Quantum Cryptography (2024-04-11)