Network Working Group
Internet-Draft
Obsoletes: 8756 (if approved)
Intended status: Informational
Expires: 4 October 2026

M. Jenkins
NSA-CCSS
2 April 2026

Commercial National Security Algorithm (CNSA) Suite 2.0 Profile for Certificate Management over CMS

draft-jenkins-cnsa2-cmc-profile-02

Abstract

This document specifies a profile of the Certificate Management over CMS (CMC) protocol for managing X.509 public key certificates in applications that use the Commercial National Security Algorithm (CNSA) Suite published by the United States Government.

The profile applies to the capabilities, configuration, and operation of all components of US National Security Systems that manage X.509 public key certificates over CMS. It is also appropriate for all other US Government systems that process high-value information.

This memo is not an IETF standard, and does not represent IETF community consensus. The profile is made publicly available here for use by developers and operators of these and any other system deployments. This document obsoletes [RFC8756], the CNSA 1.0 guidance.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies a profile of the Certificate Management over CMS (CMC) protocol to comply with the United States National Security Agency’s Commercial National Security Algorithm (CNSA) Suite [cnsafaq]. The profile applies to the capabilities, configuration, and operation of all components of US National Security Systems that employ managed X.509 certificates. US National Security Systems are described in NIST Special Publication 800-59 [SP80059]. The profile is also appropriate for all other US Government systems that process high-value information. It is made publicly available for use by developers and operators of these and any other system deployments.

This document does not define any new cryptographic algorithm; instead, it defines a CNSA-compliant profile of CMC. CMC is defined in [RFC5272], [RFC5273], and [RFC5274] and is updated by [RFC6402]. This document profiles CMC to manage X.509 public key certificates in compliance with the CNSA Suite Certificate and Certificate Revocation List (CRL) profile [I-D.jenkins-cnsa2-pkix-profile]. This document specifically focuses on defining CMC interactions for both the initial enrollment and rekey of CNSA Suite public key certificates between a client and a Certification Authority (CA). One or more Registration Authorities (RAs) may act as intermediaries between the client and the CA. Non-requirement aspects of this profile may be further tailored by specific communities to meet their needs. Specific communities will also define certificate policies that implementations need to comply with.

This memo is not an IETF standard, and does not represent IETF community consensus.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The terminology in [RFC5272] Section 2.1 applies to this profile.

The term "certificate request" is used to refer to a single PKCS #10 or Certificate Request Message Format (CRMF) structure. All PKI Requests are Full PKI Requests, and all PKI Responses are Full PKI Responses; the respective set of terms should be interpreted synonymously in this document.

3. The Commercial National Security Algorithm Suite

The National Security Agency (NSA) profiles commercial cryptographic algorithms and protocols as part of its mission to support secure, interoperable communications for US Government National Security Systems. To this end, it publishes guidance both to assist with transitioning the United States Government to new algorithms and to provide vendors, and the Internet community in general, with information concerning their proper use and configuration within the scope of US Government National Security Systems (NSS).

The Commercial National Security Algorithm (CNSA) Suite is the set of approved commercial algorithms that can be used by vendors and IT users to meet cybersecurity and interoperability requirements for NSS. The first suite of CNSA Suite algorithms, “Suite B”, established a baseline for use of commercial algorithms to protect classified information. The next suite, “CNSA 1.0”, served as a bridge between the original set and a fully post-quantum cryptographic capability. The current suite, “CNSA 2.0”, establishes fully PQ protection [cnsafaq].

The National Institute for Standards and Technology (NIST) has standardized several post-quantum asymmetric algorithms. From these, NSA has selected two: one for signing ML-DSA-87, and another for key establishment ML-KEM-1024. With SHA-384 (or SHA-512), AES-256, and LMS/XMSS, these comprise the CNSA Suite 2.0.

The NSA is authoring a set of RFCs, including this one, to provide updated guidance concerning the use of certain commonly available commercial algorithms in IETF protocols. These RFCs can be used in conjunction with other RFCs and cryptographic guidance (e.g., NIST Special Publications) to properly protect Internet traffic and data-at-rest for US Government National Security Systems.

4. General Requirements

This document assumes that the required trust anchors have been securely provisioned to the client and, when applicable, to any RAs.

All requirements in [RFC5272], [RFC5273], [RFC5274], and [RFC6402] apply, except where overridden by this profile.

The term "client" in this profile typically refers to an end-entity. However, it may instead refer to a third party acting on the end-entity's behalf. The client may or may not be the entity that actually generates the key pair, but it does perform the CMC protocol interactions with the RA and/or CA. For example, the client may be a token management system that communicates with a cryptographic token through an out-of-band secure protocol.

This profile uses the term "rekey" in the same manner as CMC does (defined in [RFC5272] Section 2). The profile makes no specific statements about the ability to do "renewal" operations; however, the statements applicable to "rekey" should be applied to "renewal" as well.

This profile may be used to manage RA and CA certificates. In that case, the terms "end-entity" and "end-entity certificate" refers to the RA and RA certificate or CA and CA certificate respectively. Note that this usage is ad hoc and is consistent with [RFC5272] but not the [RFC5280] definition which states that "End entity certificates are issued to subjects that are not authorized to issue certificates".

Signatures MUST be ML-DSA-87. The ML-DSA algorithm incorporates an internal hashing function, so there is no need to apply a hashing algorithm before signing. Where an application or implementation makes it more efficient to perform hashing externally, the external-μ mechanism described in Step 6 of Algorithm 7 of [FIPS204] and Section 8 of [RFC9881] MAY be used. Any other hashing outside of ML-DSA or ML-KEM MUST use either SHA-384 or SHA-512; SHA-384 SHOULD be used. HashML-DSA is not permitted.

5. Client Requirements: Generating PKI Requests

This section specifies the conventions employed when a client requests a certificate from a Public Key Infrastructure (PKI).

The Full PKI Request MUST be used; it MUST be encapsulated in a SignedData; and the SignedData MUST be constructed in accordance with [draft-becker-cnsa2-smime-profile-00]. The PKIData content type defined in [RFC5272] is used with the following additional requirements:

* controlSequence SHOULD be present. TransactionId and SenderNonce SHOULD be included. Other CMC controls MAY be included.

* reqSequence MUST be present. It MUST include at least one tcr (see Section 5.1) or crm (see Section 5.2) TaggedRequest. Support for the orm choice is OPTIONAL.

The private signing key used to generate the encapsulating SignedData MUST correspond to the public key of an existing signature certificate unless an appropriate signature certificate does not yet exist, such as during initial enrollment. In that case, procedural means that ensure the identity of the requestor MUST be used; see Appendix A.1.

The encapsulating SignedData MUST be generated using SHA-384 (SignerInfo digestAlgorithm, when used to compute the message-digest attribute) and ML-DSA-87 (SignerInfo signatureAlgorithm).

A Full PKI Request that is authenticated using a shared-secret (e.g. because no appropriate certificate exists yet to authenticate the request) MUST be signed using the private key corresponding to the public key of one of the requested certificates contained therein.

5.1. Tagged Certification Request

The reqSequence tcr choice conveys PKCS #10 [RFC2986] syntax. The CertificateRequest MUST comply with [RFC5272] Section 3.2.1.2.1, with the following additional requirements:

* certificationRequestInfo:

  - subjectPublicKeyInfo MUST be set as defined in [I-D.jenkins-cnsa2-pkix-profile].

  - Attributes:

    o The ExtensionReq attribute MUST be included with its contents as follows:

      + The keyUsage extension MUST be included, and it MUST be set as defined in [I-D.jenkins-cnsa2-pkix-profile].

      + For rekey requests, if the subject field of the certificate being used to generate the signature is NULL, the SubjectAltName extension MUST be included and set equal to the SubjectAltName of the certificate that is being used to sign the SignedData encapsulating the request (i.e., not the certificate being rekeyed).

      + Other extension requests MAY be included as desired.

    o The ChangeSubjectName attribute, as defined in [RFC6402], MUST be included if the Full PKI Request encapsulating this Tagged Certification Request is being signed by a key for which a certificate currently exists and the existing certificate’s subject field or SubjectAltName extension does not match the desired subject name or SubjectAltName extension of this certification request.

- signatureAlgorithm MUST be id-ml-dsa-87.

- signature MUST be generated using the private key corresponding to the public key in the CertificationRequestInfo for signature certification requests, or with an existing signing private key that has been previously certified to the same Subject.

5.2. Certificate Request Message

The reqSequence crm choice conveys Certificate Request Message Format (CRMF) [RFC4211] syntax. The CertReqMsg MUST comply with [RFC5272] Section 3.2.1.2.2, with the following additional requirements:

* popo MUST be included using the signature (POPOSigningKey) proof-of-possession choice and be set as defined in [RFC4211] Section 4.1 for signature certification requests. The POPOSigningKey poposkInput field MUST be omitted. The POPOSigningKey algorithmIdentifier MUST be id-ml-dsa-87. The signature MUST be generated using the private key corresponding to the public key in the CertTemplate, or to the public key in an existing signature certificate issued to the same Subject.

The CertTemplate MUST comply with [RFC5272] Section 3.2.1.2.2, with the following additional requirements:

* If version is included, it MUST be set to 2 as defined in [I-D.jenkins-cnsa2-pkix-profile].

* publicKey MUST be set as defined in [I-D.jenkins-cnsa2-pkix-profile].

* Extensions:

  - The keyUsage extension MUST be included, and it MUST be set as defined in [I-D.jenkins-cnsa2-pkix-profile].

  - For rekey requests, the SubjectAltName extension MUST be included and set equal to the SubjectAltName of the certificate that is being used to sign the SignedData encapsulating the request (i.e., not the certificate being rekeyed) if the subject name of the certificate being used to generate the signature is NULL.

  - Other extension requests MAY be included as desired.

* Controls:

  - The ChangeSubjectName attribute, as defined in [RFC6402], MUST be included if the Full PKI Request encapsulating this Tagged Certification Request is being signed by a key for which a certificate currently exists and the existing certificate's subject name or SubjectAltName extension does not match the desired subject name or SubjectAltName extension of this certification request.

6. RA Requirements

This section addresses the optional case where one or more RAs act as intermediaries between clients and a CA as described in [RFC5272] Section 7. In this section, the term "client" refers to the entity from which the RA received the PKI Request. This section is only applicable to RAs.

6.1. RA Processing of Requests

RAs conforming to this document MUST ensure that only the permitted signature and hash algorithms described throughout this profile are used in requests; if they are not, the RA MUST reject those requests. The RA SHOULD return a CMCFailInfo with the value of badAlg [RFC5272].

When processing end-entity-generated SignedData objects, RAs MUST NOT perform Cryptographic Message Syntax (CMS) Content Constraints (CCC) certificate extension processing [RFC6010].

Other RA processing is performed as described in [RFC5272].

6.2. RA-Generated PKI Requests

RAs mediate the certificate request process by collecting client requests in batches. The RA MUST encapsulate client-generated PKI Requests in a new RA-signed PKI Request, it MUST create a Full PKI Request encapsulated in a SignedData, and the SignedData MUST be constructed in accordance with [draft-becker-cnsa2-smime-profile-00]. The PKIData content type complies with [RFC5272] with the following additional requirements:

* controlSequence MUST be present. It MUST include the following CMC controls: Transaction ID, Sender Nonce, and Batch Requests. Other appropriate CMC controls MAY be included.

* cmsSequence MUST be present. It contains the original, unmodified request(s) received from the client.

```
      SignedData (applied by the RA)
        PKIData
          controlSequence (Transaction ID, Sender Nonce,
                                              Batch Requests)
          cmsSequence
            SignedData (applied by client)
              PKIData
                controlSequence (Transaction ID, Sender Nonce)
                reqSequence
                  TaggedRequest
                  {TaggedRequest}
            {SignedData     (second client request)
              PKIData...}
```

Authorization to sign RA-generated Full PKI Requests SHOULD be indicated in the RA certificate by inclusion of the id-kp-cmcRA Extended Key Usage (EKU) from [RFC6402]. The RA certificate MAY also include the CCC certificate extension [RFC6010], or it MAY indicate authorization through inclusion of the CCC certificate extension alone. The RA certificate may also be authorized through the local configuration.

If the RA is authorized via the CCC extension, then the CCC extension MUST include the object identifier for the PKIData content type. CCC SHOULD be included if constraints are to be placed on the content types generated.

The outer SignedData MUST be generated using SHA-384 (SignerInfo digestAlgorithm, to compute the message-digest attribute) and ML-DSA-87 (SignerInfo signatureAlgorithm).

If the Full PKI Response is a successful response to a PKI Request that only contained a Get Certificate or Get CRL control, then the algorithm used in the response MUST match the algorithm used in the request.

6.3. RA-Generated PKI Responses

An RA that also generates responses MUST assert authority to do so by at least one of

* inclusion of the id-kp-cmcRA EKU from [RFC6402] in the RA certificate (this method is preferred).

* inclusion of include the CCC certificate extension [RFC6010] in the RA certificate, with the object identifier for the PKIResponse content type.

* assertion of authorization through a locally configured implementation-specific mechanism.

7. CA Requirements

This section specifies the requirements for CAs that receive PKI Requests and generate PKI Responses.

7.1. CA Processing of PKI Requests

CAs conforming to this document MUST ensure that only the permitted signature and hash algorithms described throughout this profile are used in requests; if they are not, the CA MUST reject those requests. The CA SHOULD return a CMCStatusInfoV2 control with a CMCStatus of failed and a CMCFailInfo with the value of badAlg [RFC5272].

For requests involving an RA (i.e., batched requests), the CA MUST verify the RA's authorization. The following certificate fields MUST NOT be modifiable using the Modify Certification Request control: publicKey and the keyUsage extension. The request MUST be rejected if an attempt to modify those certification request fields is present. The CA SHOULD return a CMCStatusInfoV2 control with a CMCStatus of failed and a CMCFailInfo with a value of badRequest.

When processing end-entity-generated SignedData objects, CAs MUST NOT perform CCC certificate extension processing [RFC6010].

If a client-generated PKI Request includes the ChangeSubjectName attribute as described in Section 5.1 or Section 5.2 above, the CA MUST ensure that name change is authorized. The mechanism for ensuring that the name change is authorized is out of scope. A CA that performs this check and finds that the name change is not authorized MUST reject the PKI Request. The CA SHOULD return an Extended CMC Status Info control (CMCStatusInfoV2) with a CMCStatus of failed.

Other processing of PKIRequests is performed as described in [RFC5272].

7.2. CA-Generated PKI Responses

CAs send PKI Responses to both client-generated requests and RA-generated requests. If a Full PKI Response is returned in direct response to a client-generated request, it MUST be encapsulated in a SignedData, and the SignedData MUST be constructed in accordance with [draft-becker-cnsa2-smime-profile-00].

If the PKI Response is in response to an RA-generated PKI Request, then the above PKI Response is encapsulated in another CA-generated PKI Response. That PKI Response MUST be encapsulated in a SignedData, and the SignedData MUST be constructed in accordance with [draft-becker-cnsa2-smime-profile-00]. The above PKI Response is placed in the encapsulating PKI Response cmsSequence field. The other fields are as above with the addition of the batch response control in controlSequence. The following illustrates a successful CA response to an RA-encapsulated PKI Request, both of which include Transaction IDs and Nonces:

```
SignedData (applied by the CA)
  PKIResponse
    controlSequence (Transaction ID, Sender Nonce, Recipient
                     Nonce, Batch Response)
    cmsSequence
      SignedData (applied by CA and includes returned
                  certificates)
        PKIResponse
          controlSequence (Transaction ID, Sender Nonce,
                           Recipient Nonce)
```

The same private key used to sign certificates MUST NOT be used to sign Full PKI Response messages. Instead, a separate certificate indicating authorization to sign CMC responses MUST be used.

Authorization to sign Full PKI Responses SHOULD be indicated in the CA certificate by inclusion of the id-kp-cmcCA EKU from [RFC6402]. The CA certificate MAY also include the CCC certificate extension [RFC6010], or it MAY indicate authorization through inclusion of the CCC certificate extension alone. The CA certificate may also be authorized through local configuration.

In order for a CA certificate using the CCC certificate extension to be authorized to generate responses, the object identifier for the PKIResponse content type must be present in the CCC certificate extension. CCC SHOULD be included if constraints are to be placed on the content types generated.

Signatures applied to individual certificates are as required in [I-D.jenkins-cnsa2-pkix-profile].

8. Client Requirements: Processing PKI Responses

Clients conforming to this document MUST ensure that only the permitted signature and hash algorithms described throughout this profile are used in responses; if they are not, the client MUST reject those responses.

Clients MUST authenticate all Full PKI Responses. This includes verifying that the PKI Response is signed by an authorized CA or RA whose certificate validates back to a trust anchor. The client MUST verify that

* the CA certificate includes the id-kp-cmcCA EKU or a CCC extension asserting the PKIResponse content type, or

* the CA is authorized to sign responses through a locally configured implementation-specific mechanism.

The PKI Response can be signed by an RA if it is an error message, if it is a response to a Get Certificate or Get CRL request, or if the PKI Response contains an inner PKI Response signed by a CA. In the last case, each layer of PKI Response MUST still contain an authorized, valid signature signed by an entity with a valid certificate that verifies back to an acceptable trust anchor. The client MUST verify that

* the RA certificate includes the id-kp-cmcRA EKU or a CCC extension that includes the object identifier for the PKIResponse content type, or

* the RA is authorized to sign responses through a local configured implementation-specific mechanism.

When a newly issued certificate is included in the PKI Response, the client MUST verify that the newly issued certificate’s public key matches the public key that the client requested. The client MUST also ensure that the certificate’s signature is valid and that the signature validates back to an acceptable trust anchor.

Clients MUST reject PKI Responses that do not pass these tests, and document the rejection in a way appropriate to the system. For example, the client MAY construct a CMC Status Info control (CMCStatusInfoV2) with the CMC Stuats set to "failed", and display the code to a user console, append to an error log, or communicate to a server, depending on local policy. Local policy will determine whether the client returns a Full PKI Response with an Extended CMC Status Info control (CMCStatusInfoV2) with the CMCStatus set to failed to a user console, error log, or the server.

If the Full PKI Response contains an Extended CMC Status Info control with a CMCStatus set to failed, then local policy will determine whether the client resends a duplicate certification request back to the server or an error state is returned to a console or error log.

9. Security Considerations

Protocol security considerations are found in [RFC2986], [RFC4211], [draft-becker-cnsa2-smime-profile-00], [RFC5272], [RFC5273], [RFC5274], [I-D.jenkins-cnsa2-pkix-profile], and [RFC6402]. When CCC is used to authorize RA and CA certificates, then the security considerations in [RFC6010] also apply. Algorithm security considerations are found in [draft-becker-cnsa2-smime-profile-00].

This specification requires implementations to generate key pairs and other random values. The use of inadequate pseudorandom number generators (PRNGs) can result in little or no security. The generation of quality random numbers is difficult. NIST Special Publication 800-90A [SP80090A], FIPS 186 [FIPS186], and [RFC4086] offer random number generation guidance.

As is the case with all digital signature schemes, and especially those employed in security infrastructure, protection of private keys is paramount. Where possible, security tokens (e.g. HSMs) should be used to mitigate the risk of key compromise.

When RAs are used, the list of authorized RAs MUST be securely distributed out of band to CAs.

Presence of the POP Link Witness Version 2 and POP Link Random attributes protects against substitution attacks.

The certificate policy for a particular environment will specify whether expired certificates can be used to sign certification requests.

10. IANA Considerations

This document has no IANA actions.

11. References

11.1. Normative References

[draft-becker-cnsa2-smime-profile-00]
Jenkins, M. and A. Becker, "Commercial National Security Algorithm (CNSA) Suite Profile for Secure/ Multipurpose Internet Mail Extensions (S/MIME)", March 2025, <https://datatracker.ietf.org/doc/draft-becker-cnsa2-smime-profile/>.

[I-D.jenkins-cnsa2-pkix-profile]
Jenkins, M. and A. Becker, "Commercial National Security Algorithm Suite Certificate and Certificate Revocation List Profile", January 2025, <https://datatracker.ietf.org/doc/draft-jenkins-cnsa2-pkix-profile/>.

[cnsafaq] National Security Agency, "The Commercial National Security Algorithm Suite 2.0 and Quantum Computing FAQ", December 2024, <https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF>.

[FIPS186] National Institute of Standards and Technology, "Digital Signature Standard (DSS)", DOI 10.6028/NIST.FIPS.186-5, FIPS PUB 186-5, July 2013, <http://doi.org/10.6028/NIST.FIPS.186-5.pdf>.

[FIPS204] National Institute of Standards and Technology (2024), "Module-Lattice-Based Digital Signature Standard", (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publication (FIPS), NIST FIPS 204, <https://doi.org/10.6028/NIST.FIPS.204>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, DOI 10.17487/RFC2986, November 2000, <https://www.rfc-editor.org/info/rfc2986>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, DOI 10.17487/RFC4211, September 2005, <https://www.rfc-editor.org/info/rfc4211>.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <https://www.rfc-editor.org/info/rfc5272>.

[RFC5273] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC): Transport Protocols", RFC 5273, DOI 10.17487/RFC5273, June 2008, <https://www.rfc-editor.org/info/rfc5273>.

[RFC5274] Schaad, J. and M. Myers, "Certificate Management Messages over CMS (CMC): Compliance Requirements", RFC 5274, DOI 10.17487/RFC5274, June 2008, <https://www.rfc-editor.org/info/rfc5274>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6010] Housley, R., Ashmore, S., and C. Wallace, "Cryptographic Message Syntax (CMS) Content Constraints Extension", RFC 6010, DOI 10.17487/RFC6010, September 2010, <https://www.rfc-editor.org/info/rfc6010>.

[RFC6402] Schaad, J., "Certificate Management over CMS (CMC) Updates", RFC 6402, DOI 10.17487/RFC6402, November 2011, <https://www.rfc-editor.org/info/rfc6402>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8756] Jenkins, M. and L. Zieglar, "Commercial National Security Algorithm (CNSA) Suite Profile of Certificate Management over CMS", RFC 8756, DOI 10.17487/RFC8756, March 2020, <https://www.rfc-editor.org/info/rfc8756>.

[RFC9881] Massimo, J., Kampanakis, P., Turner, S., and B. E. Westerbaan, "Internet X.509 Public Key Infrastructure -- Algorithm Identifiers for the Module-Lattice-Based Digital Signature Algorithm (ML-DSA)", RFC 9881, DOI 10.17487/RFC9881, October 2025, <https://www.rfc-editor.org/info/rfc9881>.

[RFC9935] Turner, S., Kampanakis, P., Massimo, J., and B. E. Westerbaan, "Internet X.509 Public Key Infrastructure - Algorithm Identifiers for the Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM)", RFC 9935, DOI 10.17487/RFC9935, March 2026, <https://www.rfc-editor.org/info/rfc9935>.

11.2. Informative References

[SP80059] National Institute of Standards and Technology, "Guideline for Identifying an Information System as a National Security System", DOI 10.6028/NIST.SP.800-59, Special Publication 800-59, August 2003, <https://csrc.nist.gov/publications/detail/sp/800-59/final>.

[SP80090A] National Institute of Standards and Technology, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators", DOI 10.6028/NIST.SP.800-90Ar1, Special Publication 800-90A Revision 1, June 2015, <http://doi.org/10.6028/NIST.SP.800-90Ar1>.

Appendix A. Scenarios

This section illustrates several potential certificate enrollment and rekey scenarios supported by this profile. This section does not intend to place any limits or restrictions on the use of CMC.

A.1. Initial Enrollment

This section describes three scenarios for authenticating initial enrollment requests:

1. Previously certified signature key-pair (e.g., Manufacturer Installed Certificate).

2. Shared-secret distributed securely out of band.

3. RA authentication.

A.1.1. Previously Certified Signature Key-Pair

In this scenario, the end-entity has a private signing key and a corresponding public key certificate obtained from a cryptographic module manufacturer recognized by the CA. The end-entity signs a Full PKI Request with the private key that corresponds to the subject public key of the previously installed signature certificate. The CA will verify the authorization of the previously installed certificate and issue an appropriate new certificate to the end-entity.

A.1.2. Shared-Secret Distributed Securely Out of Band

In this scenario, the CA distributes a shared-secret out of band to the end-entity that the end-entity uses to authenticate its certification request. The end-entity signs the Full PKI Request with the private key for which the certification is being requested. The end-entity includes the Identity Proof Version 2 control to authenticate the request using the shared-secret. The CA uses either the Identification control or the subject name in the end-entity’s enclosed PKCS #10 [RFC2986] or CRMF [RFC4211] certification request message to identify the request. The end-entity performs either the POP Link Witness Version 2 mechanism as described in [RFC5272] Section 6.3.1.1 or the shared-secret/subject distinguished name linking mechanism as described in [RFC5272] Section 6.3.2. The subject name in the enclosed PKCS #10 [RFC2986] or CRMF [RFC4211] certification request does not necessarily match the issued certificate, as it may be used just to help identify the request (and the corresponding shared-secret) to the CA.

A.1.3. RA Authentication

In this scenario, the end-entity does not automatically authenticate its enrollment request to the CA, either because the end-entity has nothing to authenticate the request with or because the organizational policy requires an RA’s involvement. The end-entity creates a Full PKI Request and sends it to an RA. The RA verifies the authenticity of the request. If the request is approved, the RA encapsulates and signs the request as described in Section 5.2, forwarding the new request on to the CA. The subject name in the PKCS #10 [RFC2986] or CRMF [RFC4211] certification request is not required to match the issued certificate; it may be used just to help identify the request to the RA and/or CA.

A.2. Rekey

There are two scenarios to support the rekey of certificates that are already enrolled. One addresses the rekey of signature certificates, and the other addresses the rekey of key establishment certificates. Typically, organizational policy will require certificates to be currently valid to be rekeyed, and it may require initial enrollment to be repeated when rekey is not possible. However, some organizational policies might allow a grace period during which an expired certificate could be used to rekey.

A.2.1. Rekey of Signature Certificates

When a signature certificate is rekeyed, the PKCS #10 [RFC2986] or CRMF [RFC4211] certification request message enclosed in the Full PKI Request will include the same subject name as the current signature certificate. The Full PKI Request will be signed by the current private key corresponding to the current signature certificate.

A.2.2. Rekey of Key Establishment Certificates

When a key establishment certificate is rekeyed, the Full PKI Request will generally be signed by the current private key corresponding to the current signature certificate. If there is no current signature certificate, one of the initial enrollment options in Appendix A.1 may be used.

Author’s Address

Michael Jenkins
National Security Agency
Email: mjjenki@cyber.nsa.gov