Stream: Independent Submission
RFC: 8756
Category: Informational
Published: March 2020
ISSN: 2070-1721
Authors: M. Jenkins, *NSA*; L. Zieglar, *NSA*

RFC 8756
Commercial National Security Algorithm (CNSA) Suite Profile of Certificate Management over CMS

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.

The profile is made publicly available here for use by developers and operators of these and any other system deployments.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
   1.1. Terminology
2. The Commercial National Security Algorithm Suite
3. Requirements and Assumptions
4. Client Requirements: Generating PKI Requests
   4.1. Tagged Certification Request
   4.2. Certificate Request Message
5. RA Requirements
   5.1. RA Processing of Requests
   5.2. RA-Generated PKI Requests
   5.3. RA-Generated PKI Responses
6. CA Requirements
   6.1. CA Processing of PKI Requests
   6.2. CA-Generated PKI Responses
7. Client Requirements: Processing PKI Responses
8. Shared-Secrets
9. Security Considerations
10. IANA Considerations
11. References
   11.1. Normative References
   11.2. Informative References
Appendix A. Scenarios
   A.1. Initial Enrollment
   A.2. Rekey
Authors' Addresses

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 [CNSA]. The profile applies to the capabilities, configuration, and operation of all components of US National Security Systems [SP80059]. It 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 suites; 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 [RFC8603]. 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. 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.

1.1. 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.

2. 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 the US Government transition 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.

Recently, cryptographic transition plans have become overshadowed by the prospect of the development of a cryptographically relevant quantum computer. The NSA has established the Commercial National Security Algorithm (CNSA) Suite to provide vendors and IT users near-term flexibility in meeting their cybersecurity interoperability requirements. The purpose behind this flexibility is to avoid having vendors and customers make two major transitions in a relatively short timeframe, as we anticipate a need to shift to quantum-resistant cryptography in the near future.

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.

3. Requirements and Assumptions

Elliptic Curve Digital Signature Algorithm (ECDSA) and Elliptic Curve Diffie-Hellman (ECDH) key pairs are on the P-384 curve. FIPS 186-4 [FIPS186], Appendix B.4 provides useful guidance for elliptic curve key pair generation that **SHOULD** be followed by systems that conform to this document.

RSA key pairs (public, private) are identified by the modulus size expressed in bits; RSA-3072 and RSA-4096 are computed using moduli of 3072 bits and 4096 bits, respectively.

RSA signature key pairs used in CNSA Suite-compliant implementations are either RSA-3072 or RSA-4096. The RSA exponent e **MUST** satisfy $2^{16} < e < 2^{256}$ and be odd per [FIPS186].

It is recognized that, while the vast majority of RSA signatures are currently made using the RSASSA-PKCS1-v1_5 algorithm, the preferred RSA signature scheme for new applications is RSASSA-PSS. CNSA Suite-compliant X.509 certificates will be issued in accordance with [RFC8603], and while those certificates must be signed and validated using RSASSA-PKCS1-v1_5, the subject's private key can be used to generate signatures of either signing scheme. Where use of RSASSA-PSS is indicated in this document, the following parameters apply:

- The hash algorithm **MUST** be id-sha384 as defined in [RFC8017];
- The mask generation function **MUST** use the algorithm identifier mfg1SHA384Identifier as defined in [RFC4055];
- The salt length **MUST** be 48 octets; and
- The trailerField **MUST** have value 1.

These parameters will not appear in a certificate and **MUST** be securely communicated with the signature, as required by Section 2.2 of [RFC4056]. Application developers are obliged to ensure that the chosen signature scheme is appropriate for the application and will be interoperable within the intended operating scope of the application.

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.

This profile was developed with the scenarios described in Appendix A in mind. However, use of this profile is not limited to just those scenarios.

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 Section 2 of [RFC5272]). 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/or CA certificates. In that case, the RA and/or CA whose certificate is being managed is considered to be the end-entity.

This profile does not discuss key establishment certification requests from cryptographic modules that cannot generate a one-time signature with a key establishment key for proof-of-possession purposes. In that case, a separate profile would be needed to define the use of another proof-of-possession technique.

4. 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 [RFC8755]. 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.
  - If the request is being authenticated using a shared-secret, then Identity Proof Version 2 control **MUST** be included with the following constraints:
    - hashAlgId **MUST** be id-sha384 for all certification requests (algorithm OIDs are defined in [RFC5754]).
    - macAlgId **MUST** be HMAC-SHA384 (the Hashed Message Authentication Code (HMAC) algorithm is defined in [RFC4231]).
  - If the subject name included in the certification request is NULL or otherwise does not uniquely identify the end-entity, then the POP Link Random control **MUST** be included, and the POP Link Witness Version 2 control **MUST** be included in the inner PKCS #10 [RFC2986] or Certificate Request Message Format (CRMF) [RFC4211] request as described in Sections 4.1 and 4.2.

- reqSequence **MUST** be present. It **MUST** include at least one tcr (see Section 4.1) or crm (see Section 4.2) 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.

The encapsulating SignedData **MUST** be generated using SHA-384 and either ECDSA on P-384 or RSA using either RSASSA-PKCS1-v1_5 or RSASSA-PSS with an RSA-3072 or RSA-4096 key.

If an appropriate signature certificate does not yet exist and if a Full PKI Request includes one or more certification requests and is authenticated using a shared-secret (because no appropriate certificate exists yet to authenticate the request), the Full PKI Request **MUST** be signed using the private key corresponding to the public key of one of the requested certificates. When necessary (i.e., because there is no existing signature certificate and there is no signature certification request included), a Full PKI Request **MAY** be signed using a key pair intended for use in a key establishment certificate. However, servers are not required to allow this behavior.

4.1. 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 Section 5.4 of [RFC8603].
  - Attributes:
    - 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 [RFC8603].
      - 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 field of the certificate being used to generate the signature is NULL.
      - Other extension requests **MAY** be included as desired.
    - 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.
    - The POP Link Witness Version 2 attribute **MUST** be included if the request is being authenticated using a shared-secret and the subject name in the certification request is NULL or otherwise does not uniquely identify the end-entity. In the POP Link Witness Version 2 attribute, keyGenAlgorithm **MUST** be id-sha384 for certification requests, as defined in [RFC5754]; macAlgorithm **MUST** be HMAC-SHA384, as defined in [RFC4231].

  - signatureAlgorithm **MUST** be ecdsa-with-sha384 for P-384 certification requests and sha384WithRSAEncryption or id-RSASSA-PSS for RSA-3072 and RSA-4096 certification requests.
  - signature **MUST** be generated using the private key corresponding to the public key in the CertificationRequestInfo for both signature and key establishment certification requests. The signature provides proof-of-possession of the private key to the CA.

4.2. 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 both signature and key establishment certification requests. The POPOSigningKey poposkInput field **MUST** be omitted. The POPOSigningKey algorithmIdentifier **MUST** be ecdsa-with-sha384 for P-384 certification requests and sha384WithRSAEncryption or id-RSASSA-PSS for RSA-3072 and RSA-4096 certification requests. The signature **MUST** be generated using the private key corresponding to the public key in the CertTemplate.

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 Section 5.3 of [RFC8603].
- publicKey **MUST** be set as defined in Section 5.4 of [RFC8603].
- Extensions:
  - The keyUsage extension **MUST** be included, and it **MUST** be set as defined in [RFC8603].
  - 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.
  - The POP Link Witness Version 2 attribute **MUST** be included if the request is being authenticated using a shared-secret and the subject name in the certification request is NULL or otherwise does not uniquely identify the end-entity. In the POP Link Witness

Version 2 attribute, keyGenAlgorithm **MUST** be id-sha384 for certification requests; macAlgorithm **MUST** be HMAC-SHA384 when keyGenAlgorithm is id-sha384.

5. RA Requirements

This section addresses the optional case where one or more RAs act as intermediaries between clients and a CA as described in Section 7 of [RFC5272]. 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.

5.1. RA Processing of Requests

RAs conforming to this document **MUST** ensure that only the permitted signature, hash, and MAC 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].

5.2. 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 [RFC8755]. 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 and either ECDSA on P-384 or RSA using RSASSA-PKCS1-v1_5 or RSASSA-PSS with an RSA-3072 or RSA-4096 key.

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.

5.3. RA-Generated PKI Responses

In order for an RA 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.

6. CA Requirements

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

6.1. CA Processing of PKI Requests

CAs conforming to this document **MUST** ensure that only the permitted signature, hash, and MAC 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 4.1 or 4.2 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].

6.2. 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 [RFC8755].

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 [RFC8755]. 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 [RFC8603].

The signature on the SignedData of a successful response to a client-generated request, or each individual inner SignedData on the successful response to an RA-generated request, **MUST** be generated using SHA-384 and either ECDSA on P-384 or RSA using RSASSA-PKCS1-v1_5 or RSASSA-PSS with an RSA-3072 or RSA-4096 key. An unsuccessful response **MUST** be signed using the same key type and algorithm that signed the request.

The outer SignedData on the Full PKI Response to any RA-generated PKI Request **MUST** be signed with the same key type and algorithm that signed the request.

The SignedData on a successful Full PKI Response to a PKI Request that only contained a Get Certificate or Get CRL control **MUST** be signed with the same key type and algorithm that signed the request.

7. Client Requirements: Processing PKI Responses

Clients conforming to this document **MUST** ensure that only the permitted signature, hash, and MAC 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 authorized CA certificate **MUST** include the id-kp-cmcCA EKU and/or a CCC extension that includes the object identifier for the PKIResponse content type. Otherwise, the CA is determined to be authorized to sign responses through an 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 authorized RA certificate **MUST** include the id-kp-cmcRA EKU and/or include a CCC extension that includes the object identifier for the PKIResponse content type. Otherwise, the RA is determined to be authorized to sign responses through local configuration.

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. 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.

8. Shared-Secrets

When the Identity Proof V2 and POP Link Witness V2 controls are used, the shared-secret **MUST** be randomly generated and securely distributed. The shared-secret **MUST** provide at least 192 bits of strength.

9. Security Considerations

Protocol security considerations are found in [RFC2986], [RFC4211], [RFC8755], [RFC5272], [RFC5273], [RFC5274], [RFC8603], 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 [RFC8755].

Compliant with NIST Special Publication 800-57 [SP80057], this profile defines proof-of-possession of a key establishment private key by performing a digital signature. Except for one-time proof-of-possession, a single key pair **MUST NOT** be used for both signature and key establishment.

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 186-3 [FIPS186], and [RFC4086] offer random number generation guidance.

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

[CNSA] Committee on National Security Systems, "Use of Public Standards for Secure Information Sharing", CNSS Policy 15, October 2016, <https://www.cnss.gov/CNSS/issuances/Policies.cfm>.

[FIPS186]

National Institute of Standards and Technology, "Digital Signature Standard (DSS)", DOI 10.6028/NIST.FIPS.186-4, FIPS PUB 186-4, July 2013, <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>.

[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>.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, DOI 10.17487/RFC4055, June 2005, <https://www.rfc-editor.org/info/rfc4055>.

[RFC4056] Schaad, J., "Use of the RSASSA-PSS Signature Algorithm in Cryptographic Message Syntax (CMS)", RFC 4056, DOI 10.17487/RFC4056, June 2005, <https://www.rfc-editor.org/info/rfc4056>.

[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>.

[RFC4231] Nystrom, M., "Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512", RFC 4231, DOI 10.17487/RFC4231, December 2005, <https://www.rfc-editor.org/info/rfc4231>.

[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>.

[RFC5754] Turner, S., "Using SHA2 Algorithms with Cryptographic Message Syntax", RFC 5754, DOI 10.17487/RFC5754, January 2010, <https://www.rfc-editor.org/info/rfc5754>.

[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>.

[RFC8017] Moriarty, K., Ed., Kaliski, B., Jonsson, J., and A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2", RFC 8017, DOI 10.17487/RFC8017, November 2016, <https://www.rfc-editor.org/info/rfc8017>.

[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>.

[RFC8603] Jenkins, M. and L. Zieglar, "Commercial National Security Algorithm (CNSA) Suite Certificate and Certificate Revocation List (CRL) Profile", RFC 8603, DOI 10.17487/RFC8603, May 2019, <https://www.rfc-editor.org/info/rfc8603>.

[RFC8755] Jenkins, M., "Using Commercial National Security Algorithm Suite Algorithms in Secure/Multipurpose Internet Mail Extensions", RFC 8755, DOI 10.17487/RFC8755, March 2020, <https://www.rfc-editor.org/info/rfc8755>.

11.2. Informative References

[SP80057] National Institute of Standards and Technology, "Recommendation for Key Management, Part 1: General", DOI 10.6028/NIST.SP.800-57pt1r4, Special Publication 800-57, Part 1, Revision 4, January 2016, <http://doi.org/10.6028/NIST.SP.800-57pt1r4>.

[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 4.2, 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.

Authors' Addresses

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

Lydia Zieglar
National Security Agency
Email: llziegl@tycho.ncsc.mil