Certificate Management Messages over CMS (CMC): Compliance Requirements

Abstract

This document provides a set of compliance statements about the CMC (Certificate Management over CMS) enrollment protocol. The ASN.1 structures and the transport mechanisms for the CMC enrollment protocol are covered in other documents. This document provides the information needed to make a compliant version of CMC.

This document obsoletes RFCs 5274 and 6402.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-mandel-lamps-rfc5274bis/.

Discussion of this document takes place on the WG LAMPS mailing list (mailto:spasm@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/spasm/. Subscribe at https://www.ietf.org/mailman/listinfo/spasm/.

Source for this draft and an issue tracker can be found at https://github.com/TBD.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents
1. Introduction

The CMC (Certificate Management over CMS) protocol is designed in terms of a client/server relationship. In the simplest case, the client is the requestor of the certificate (i.e., the End Entity (EE)) and the server is the issuer of the certificate (i.e., the Certification Authority (CA)). The introduction of a Registration
Authority (RA) into the set of agents complicates the picture only slightly. The RA becomes the server with respect to the certificate requestor, and it becomes the client with respect to the certificate issuer. Any number of RAs can be inserted into the picture in this manner.

The RAs may serve specialized purposes that are not currently covered by this document. One such purpose would be a Key Escrow agent. As such, all certificate requests for encryption keys would be directed through this RA and it would take appropriate action to do the key archival. Key recovery requests could be defined in the CMC methodology allowing for the Key Escrow agent to perform that operation acting as the final server in the chain of agents.

If there are multiple RAs in the system, it is considered normal that not all RAs will see all certificate requests. The routing between the RAs may be dependent on the content of the certificate requests involved.

This document is divided into six sections, each section specifying the requirements that are specific to a class of agents in the CMC model. These are 1) All agents, 2) all servers, 3) all clients, 4) all End-Entities, 5) all Registration Entities, 6) all Certificate Authorities.

This document obsoletes [CMC-COMPv1] and [CMC-Updates].

2. Terminology

There are several different terms, abbreviations, and acronyms used in this document that we define here for convenience and consistency of usage:

**End-Entity (EE):** Refers to the entity that owns a key pair and for whom a certificate is issued.

**Registration Authority (RA) or Local RA (LRA):** Refers to an entity that acts as an intermediary between the EE and the CA. Multiple RAs can exist between the End-Entity and the Certification Authority. RAs may perform additional services such as key
Certification Authority (CA): Refers to the entity that issues certificates.

Client: Refers to an entity that creates a PKI Request. In this document, both RAs and EEs can be clients.

Server: Refers to the entities that process PKI Requests and create PKI Responses. In this document both CAs and RAs can be servers.

PKCS #10: Refers to the Public Key Cryptography Standard #10 [PKCS10], which defines a certification request syntax.

CRMF: Refers to the Certificate Request Message Format RFC [CRMF]. CMC uses this certification request syntax defined in this document as part of the protocol.

CMS: Refers to the Cryptographic Message Syntax RFC [CMS]. This document provides for basic cryptographic services including encryption and signing with and without key management.

PKI Request/Response: Refers to the requests/responses described in this document. PKI Requests include certification requests, revocation requests, etc. PKI Responses include certs-only messages, failure messages, etc.

Proof-of-Identity: Refers to the client proving they are who they say that they are to the server.

Proof-of-Possession (POP): Refers to a value that can be used to prove that the private key corresponding to a public key is in the possession and can be used by an end-entity.

Transport wrapper: Refers to the outermost CMS wrapping layer.

3. Requirements 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.

4. Changes since RFC 5274 and 6402

Note: For now, this section will be list of the changes introduced by each version. After WGLC, this section will be finalized.
5. Requirements for All Entities

All [CMC-STRUCT] and [CMC-TRANS] compliance statements MUST be adhered to unless specifically stated otherwise in this document.

All entities MUST support Full PKI Requests, Simple PKI Responses, and Full PKI Responses. Servers SHOULD support Simple PKI Requests.

All entities MUST support the use of the CRMF syntax for certification requests. Support for the PKCS #10 syntax for certification requests SHOULD be implemented by servers.

The extendedFailInfo field SHOULD NOT be populated in the CMCStatusInfoV2 object; the failInfo field SHOULD be used to relay this information. If the extendedFailInfo field is used, it is suggested that an additional CMCStatusInfoV2 item exist for the same body part with a failInfo field.

All entities MUST implement the HTTP transport mechanism as defined in [CMC-TRANS]. Other transport mechanisms MAY be implemented.

5.1. Cryptographic Algorithm Requirements

All entities MUST verify DSA-SHA1 and RSA-SHA1 signatures in SignedData (see [CMS-ALG]). Entities MAY verify other signature algorithms. It is strongly suggested that RSA-PSS with SHA-1 be verified (see [CMS-RSA-PSS]). It is strongly suggested that SHA-256 using RSA and RSA-PSS be verified (see [RSA-256]).
All entities **MUST** generate either DSA-SHA1 or RSA-SHA1 signatures for SignedData (see [CMS-ALG]). Other signatures algorithms **MAY** be used for generation.

All entities **MUST** support Advanced Encryption Standard (AES) as the content encryption algorithm for EnvelopedData (see [CMS-AES]). Other content encryption algorithms **MAY** be implemented.

All entities **MUST** support RSA as a key transport algorithm for EnvelopedData (see [CMS-ALG]). All entities **SHOULD** support RSA-OAEP (see [CMS-RSA-OAEP]) as a key transport algorithm. Other key transport algorithms **MAY** be implemented.

If an entity supports key agreement for EnvelopedData, it **MUST** support Diffie-Hellman (see [CMS-DH]).

If an entity supports PasswordRecipientInfo for EnvelopedData or AuthenticatedData, it **MUST** support PBKDF2 [PBKDF2] for key derivation algorithms. It **MUST** support AES key wrap (see [AES-WRAP]) as the key encryption algorithm.

If AuthenticatedData is supported, PasswordRecipientInfo **MUST** be supported.

Algorithm requirements for the Identity Proof Version 2 control Section 6.2.1 of [CMC-STRUCT] are: SHA-1 **MUST** be implemented for hashAlgId. SHA-256 **SHOULD** be implemented for hashAlgId. HMAC-SHA1 **MUST** be implemented for macAlgId. HMAC-SHA256 **SHOULD** be implemented for macAlgId.

Algorithm requirements for the Pop Link Witness Version 2 control Section 6.3.1 of [CMC-STRUCT] are: SHA-1 **MUST** be implemented for keyGenAlgorithm. SHA-256 **SHOULD** be implemented for keyGenAlgorithm. PBKDF2 [PBKDF2] **MAY** be implemented for keyGenAlgorithm. HMAC-SHA1 **MUST** be implemented for macAlgorithm. HMAC-SHA256 **SHOULD** be implemented for macAlgorithm.

Algorithm requirements for the Encrypted POP and Decrypted POP controls Section 6.7 of [CMC-STRUCT] are: SHA-1 **MUST** be implemented for witnessAlgID. SHA-256 **SHOULD** be implemented for witnessAlgID. HMAC-SHA1 **MUST** be implemented for thePOPAlgID. HMAC-SHA256 **SHOULD** be implemented for thePOPAlgID.

Algorithm requirements for Publish Trust Anchors control Section 6.15 of [CMC-STRUCT] are: SHA-1 **MUST** be implemented for hashAlgorithm. SHA-256 **SHOULD** be implemented for hashAlgorithm.

If an EE generates DH keys for certification, it **MUST** support Section 4 of [DH-POP]. EEs **MAY** support Section 3 of [DH-POP]. CAs
and RAs that do POP verification **MUST** support Section 4 of [DH-POP] and **SHOULD** support Section 3 of [DH-POP].

EEs that need to use a signature algorithm for keys that cannot produce a signature **MUST** support Appendix C of [CMC-STRUCT] and **MUST** support the Encrypted/Decrypted POP controls. CAs and RAs that do POP verification **MUST** support this signature algorithm and **MUST** support the Encrypted/Decrypted POP controls.

### 5.2. Controls

The following table lists the name and level of support required for each control.

<table>
<thead>
<tr>
<th>Control</th>
<th>EE</th>
<th>RA</th>
<th>CA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended CMC Status Info</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>CMC Status Info</td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>Identity Proof Version 2</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Identity Proof</td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>Identification</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>POP Link Random</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>POP Link Witness Version 2</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>POP Link Witness</td>
<td><strong>SHOULD</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Data Return</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Modify Cert Request</td>
<td>N/A</td>
<td><strong>MUST</strong></td>
<td>(2)</td>
</tr>
<tr>
<td>Add Extensions</td>
<td>N/A</td>
<td><strong>MAY</strong></td>
<td>(1)</td>
</tr>
<tr>
<td>Transaction ID</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Sender Nonce</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Recipient Nonce</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Encrypted POP</td>
<td>(4)</td>
<td>(5)</td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>Decrypted POP</td>
<td>(4)</td>
<td>(5)</td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>RA POP Witness</td>
<td>N/A</td>
<td><strong>SHOULD</strong></td>
<td>(1)</td>
</tr>
<tr>
<td>Get Certificate</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Get CRL</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Revocation Request</td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Registration Info</td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>Response Information</td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
<td><strong>SHOULD</strong></td>
</tr>
<tr>
<td>Query Pending</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Confirm Cert. Acceptance</td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
<td><strong>MUST</strong></td>
</tr>
<tr>
<td>Publish Trust Anchors</td>
<td>(3)</td>
<td>(3)</td>
<td>(3)</td>
</tr>
<tr>
<td>Authenticate Data</td>
<td>(3)</td>
<td>(3)</td>
<td>(3)</td>
</tr>
<tr>
<td>Batch Request</td>
<td>N/A</td>
<td><strong>MUST</strong></td>
<td>(2)</td>
</tr>
<tr>
<td>Batch Responses</td>
<td>N/A</td>
<td><strong>MUST</strong></td>
<td>(2)</td>
</tr>
<tr>
<td>Publication Information</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Control Processed</td>
<td>N/A</td>
<td><strong>MUST</strong></td>
<td>(2)</td>
</tr>
<tr>
<td>RA Identity Proof Witness</td>
<td>N/A</td>
<td><strong>MUST</strong></td>
<td>(2)</td>
</tr>
<tr>
<td>Response Body</td>
<td>(6)</td>
<td>(6)</td>
<td>N/A.</td>
</tr>
</tbody>
</table>
Table 1: CMC Control Attributes

Notes:

1. CAs **SHOULD** implement this control if designed to work with RAs.

2. CAs **MUST** implement this control if designed to work with RAs.

3. Implementation is optional for these controls. We strongly suggest that they be implemented in order to populate client trust anchors.

4. EEs only need to implement this if (a) they support key agreement algorithms or (b) they need to operate in environments where the hardware keys cannot provide POP.

5. RAs **SHOULD** implement this if they implement RA POP Witness.

6. EE's **SHOULD** implement if designed to work with RAs and **MUST** implement if intended to be used in environments where RAs are used for identity validation or key generation. RAs **SHOULD** implement and validate responses for consistency.

Strong consideration should be given to implementing the Authenticate Data and Publish Trust Anchors controls as this gives a simple method for distributing trust anchors into clients without user intervention.

5.3. CRMF Feature Requirements

The following additional restrictions are placed on CRMF features:

The registration control tokens id-regCtrl-regToken and id-regCtrl-authToken **MUST NOT** be used. No specific CMC feature is used to replace these items, but generally the CMC controls identification and identityProof will perform the same service and are more specifically defined.

The control token id-regCtrl-pkiArchiveOptions **SHOULD NOT** be supported. An alternative method is under development to provide this functionality.

The behavior of id-regCtrl-oldCertID is not presently used. It is replaced by issuing the new certificate and using the id-cmc-publishCert to remove the old certificate from publication. This operation would not normally be accompanied by an immediate revocation of the old certificate; however, that can be accomplished by the id-cmc-revokeRequest control.
The id-regCtrl-protocolEncrKey is not used.

5.4. Requirements for Clients

There are no additional requirements.

6. Requirements for Servers

There are no additional requirements.

7. Requirements for EEs

If an entity implements Diffie-Hellman, it MUST implement either the DH-POP Proof-of-Possession as defined in Section 4 of [DH-POP] or the challenge-response POP controls id-cmc-encryptedPOP and id-cmc-decryptedPOP.

8. Requirements for RAs

RAs SHOULD be able to do delegated POP. RAs implementing this feature MUST implement the id-cmc-lraPOPWitness control.

All RAs MUST implement the promotion of the id-aa-cmc-unsignedData as covered in Section 3.2.3 of [CMC-STRUCT].

9. Requirements for CAs

Providing for CAs to work in an environment with RAs is strongly suggested. Implementation of such support is strongly suggested as this permits the delegation of substantial administrative interaction onto an RA rather than at the CA.

CAs MUST perform at least minimal checks on all public keys before issuing a certificate. At a minimum, a check for syntax would occur with the POP operation. Additionally, CAs SHOULD perform simple checks for known bad keys such as small subgroups for DSA-SHA1 and DH keys [SMALL-SUB-GROUP] or known bad exponents for RSA keys.

CAs MUST enforce POP checking before issuing any certificate. CAs MAY delegate the POP operation to an RA for those cases where 1) a challenge/response message pair must be used, 2) an RA performs escrow of a key and checks for POP in that manner, or 3) an unusual algorithm is used and that validation is done at the RA.

CAs SHOULD implement both the DH-POP Proof-of-Possession as defined in Section 4 of [DH-POP] and the challenge-response POP controls id-cmc-encryptedPOP and id-cmc-decryptedPOP.
10. Security Considerations

This document uses [CMC-STRUCT] and [CMC-TRANS] as building blocks to this document. The security sections of those two documents are included by reference.

Knowledge of how an entity is expected to operate is vital in determining which sections of requirements are applicable to that entity. Care needs to be taken in determining which sections apply and fully implementing the necessary code.

Cryptographic algorithms have and will be broken or weakened. Implementers and users need to check that the cryptographic algorithms listed in this document make sense from a security level. The IETF from time to time may issue documents dealing with the current state of the art. Two examples of such documents are [SMALL-SUB-GROUP] and [HASH-ATTACKS].

11. References

11.1. Normative References


[CMS-AES] Schaad, J., "Use of the Advanced Encryption Standard (AES) Encryption Algorithm in Cryptographic Message...


Profile", RFC 4055, DOI 10.17487/RFC4055, June 2005,

11.2. Informative References

[CMC-COMPv1] Schaad, J. and M. Myers, "Certificate Management
Messages over CMS (CMC): Compliance Requirements", RFC
editor.org/rfc/rfc5274>.

[CMC-Updates] Schaad, J., "Certificate Management over CMS (CMC)
Updates", RFC 6402, DOI 10.17487/RFC6402, November 2011,

[HASH-ATTACKS] Hoffman, P. and B. Schneier, "Attacks on
Cryptographic Hashes in Internet Protocols", RFC 4270,
editor.org/rfc/rfc4270>.

Request Syntax Specification Version 1.7", RFC 2986, DOI
10.17487/RFC2986, November 2000, <https://www.rfc-
editor.org/rfc/rfc2986>.

[SMALL-SUB-GROUP] Zuccherato, R., "Methods for Avoiding the "Small-
Subgroup" Attacks on the Diffie-Hellman Key Agreement

Acknowledgements

Obviously, the authors of this version of the document would like to
thank Jim Schaad and Michael Myers for their work on the previous
version of this document.

The acknowledgment from the previous version of this document
follows:

The authors and the PKIX Working Group are grateful for the
participation of Xiaoyi Liu and Jeff Weinstein in helping to author
the original versions of this document.

The authors would like to thank Brian LaMacchia for his work in
developing and writing up many of the concepts presented in this
document. The authors would also like to thank Alex Deacon and Barb
Fox for their contributions.

Contributors

Jim Schaad
August Cellars

Michael Myers
TraceRoute Security, Inc.

Authors' Addresses

Joseph Mandel
AKAYLA, Inc.

Email: joe@akyala.com

Sean Turner
sn3rd

Email: sean@sn3rd.com