LAMPS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 12, 2021

H. Brockhaus
S. Fries
D. von Oheimb
Siemens

July 11, 2020

Lightweight CMP Profile
draft-ietf-lamps-lightweight-cmp-profile-02

Abstract

The goal of this document is to facilitate interoperability and automation by profiling the Certificate Management Protocol (CMP) version 2, the related Certificate Request Message Format (CRMF) version 2, and the HTTP Transfer for the Certificate Management Protocol. It specifies a subset of CMP and CRMF focusing on typical uses cases relevant for managing certificates of devices in many industrial and IoT scenarios. To limit the overhead of certificate management for more constrained devices only the most crucial types of operations are specified as mandatory. To foster interoperability in more complex scenarios, other types of operations are specified as recommended or optional.

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 at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 12, 2021.

Table of Contents

1. Introduction

!!! The change history was moved to Appendix C !!!

This document specifies PKI management operations supporting machine-to-machine and IoT use cases. The focus lies on maximum automation and interoperable implementation of all involved PKI entities from end entities (EE) through an optional Local Registration Authority (LRA) and the RA up to the CA. The profile makes use of the concepts

and syntax specified in CMP [RFC4210], CRMF [RFC4211], HTTP transfer for CMP [RFC6712], and CMP Updates [I-D.ietf-lamps-cmp-updates]. Especially CMP and CRMF are very feature-rich standards, while only a limited subset of the specified functionality is needed in many environments. Additionally, the standards are not always precise enough on how to interpret and implement the described concepts. Therefore, this document aims at tailoring and specifying in more detail how to use these concepts to implement lightweight automated certificate management.

1.1. Motivation for profiling CMP

CMP was standardized in 1999 and is implemented in several CA products. In 2005 a completely reworked and enhanced version 2 of CMP [RFC4210] and CRMF [RFC4211] has been published followed by a document specifying a transfer mechanism for CMP messages using http [RFC6712] in 2012.

Though CMP is a very solid and capable protocol it could be used more widely. The most important reason for not more intense application of CMP appears to be that the protocol is offering a large set of features and options but being not always precise enough and leaving room for interpretation. On the one hand, this makes CMP applicable to a very wide range of scenarios, but on the other hand a full implementation of all options is unrealistic because this would take enormous effort.

Moreover, many details of the CMP protocol have been left open or have not been specified in full preciseness. The profiles specified in Appendix D and E of [RFC4210] offer some more detailed PKI management operations. But the specific needs of highly automated scenarios for a machine-to-machine communication are not covered sufficiently.

As also 3GPP and UNISIG already put across, profiling is a way of coping with the challenges mentioned above. To profile means to take advantage of the strengths of the given protocol, while explicitly narrowing down the options it provides to exactly those needed for the purpose(s) at hand and eliminating all identified ambiguities. In this way all the general and applicable aspects of the protocol can be taken over and only the peculiarities of the target scenario need to be dealt with specifically.

Doing such a profiling for a new target environment can be a high effort because the range of available options needs to be well understood and the selected options need to be consistent with each other and with the intended usage scenario. Since most industrial PKI management use cases typically have much in common it is worth

sharing this effort, which is the aim of this document. Other standardization bodies can then reference the needed PKI management operations from this document and do not need to come up with individual profiles.

1.2. Motivation for a lightweight profile for CMP

The profiles specified in Appendix D and E of CMP have been developed in particular to manage certificates of human end entities. With the evolution of distributed systems and client-server architectures, certificates for machines and applications on them have become widely used. This trend has strengthened even more in emerging industrial and IoT scenarios. CMP is sufficiently flexible to support these very well.

Today's IT security architectures for industrial solutions typically use certificates for endpoint authentication within protocols like IPSec, TLS, or SSH. Therefore, the security of these architectures highly relies upon the security and availability of the implemented certificate management procedures.

Due to increasing security in operational networks as well as availability requirements, especially on critical infrastructures and systems with a high volume of certificates, a state-of-the-art certificate management must be constantly available and cost-efficient, which calls for high automation and reliability. The NIST Cyber Security Framework [NIST-CSFW] also refers to proper processes for issuance, management, verification, revocation, and audit for authorized devices, users and processes involving identity and credential management. Such PKI operation according to commonly accepted best practices is also required in IEC 62443-3-3 [IEC62443-3-3] for security level 2 up to security level 4.

Further challenges in many industrial systems are network segmentation and asynchronous communication, where PKI operation is often not deployed on-site but in a more protected environment of a data center or trust center. Certificate management must be able to cope with such network architectures. CMP offers the required flexibility and functionality, namely self-contained messages, efficient polling, and support for asynchronous message transfer with end-to-end security.

1.3. Existing CMP profiles

As already stated, CMP contains profiles with mandatory and optional transactions in the Appendixes D and E of [RFC4210]. Those profiles focus on management of human user certificates and do only partly

address the specific needs for certificate management automation for unattended machine or application-oriented end entities.

[RFC4210] specifies in Appendix D the following mandatory PKI management operations (all require support of, in the meantime outdated, algorithms, e.g., SHA-1 and 3-DES; all operations may enroll up to two certificates, one for a locally generated and another optional one for a centrally generated key pair; all require use of certConf/pkiConf messages for confirmation):

- Initial registration/certification; an (uninitialized) end entity requests a (first) certificate from a CA using shared secret based message authentication. The content is similar to the PKI management operation specified in Section 4.1.4 of this document.

- Certificate request; an (initialized) end entity requests another certificate from a CA using signature or shared secret based message authentication. The content is similar to the PKI management operation specified in Section 4.1.2 of this document.

- Key update; an (initialized) end entity requests a certificate from a CA (to update the key pair and/or corresponding certificate that it already possesses) using signature or shared secret based message authentication. The content is similar to the PKI management operation specified in Section 4.1.3 of this document.

Due to the two certificates that may be enrolled and the shared secret based authentication, these PKI management operations focus more on the enrollment of human users at a PKI.

[RFC4210] specifies in Appendix E the following optional PKI management operations (all require support of, in the meantime outdated, algorithms, e.g., SHA-1 and 3-DES):

- Root CA key update; a root CA updates its key pair and produces a CA key update announcement message that can be made available (via some transport mechanism) to the relevant end entities. This operation only supports a push and no pull model. The content is similar to the PKI management operation specified in Section 4.4.3 of this document.

- Information request/response; an end entity sends a general message to the PKI requesting details that will be required for later PKI management operations. The content is similar to the PKI management operation specified in Section 4.4.4 of this document.

- Cross-certification request/response (1-way); creation of a single cross-certificate (i.e., not two at once). The requesting CA MAY choose who is responsible for publication of the cross-certificate created by the responding CA through use of the PKIPublicationInfo control.

- In-band initialization using external identity certificate (this PKI management operation may also enroll up to two certificates and requires use of certConf/pkiConf messages for confirmation as specified in Appendix D of [RFC4210]). An (uninitialized) end entity wishes to initialize into the PKI with a CA, CA-1. It uses, for authentication purposes, a pre-existing identity certificate issued by another (external) CA, CA-X. A trust relationship must already have been established between CA-1 and CA-X so that CA-1 can validate the EE identity certificate signed by CA-X. Furthermore, some mechanism must already have been established within the Personal Security Environment (PSE) of the EE that would allow it to authenticate and verify PKIMessages signed by CA-1. The content is similar to the PKI management operation specified in Section 4.1.1 of this document.

Both Appendixes focus on EE to CA/RA PKI management operations and do not address further profiling of RA to CA communication as typically used for full backend automation.

3GPP makes use of CMP [RFC4210] in its Technical Specification 133 310 [ETSI-3GPP] for automatic management of IPSec certificates in UMTS, LTE, and 5G backbone networks. Since 2010 a dedicated CMP profile for initial certificate enrollment and update operations between EE and RA/CA is specified in that document.

UNISIG has included a CMP profile for certificate enrollment in the subset 137 specifying the ETRAM/ECTS on-line key management for train control systems [UNISIG] in 2015.

Both standardization bodies use CMP [RFC4210], CRMF [RFC4211], and HTTP transfer for CMP [RFC6712] to add tailored means for automated PKI management operations for unattended machine or application-oriented end entities.

1.4. Compatibility with existing CMP profiles

The profile specified in this document is compatible with CMP [RFC4210] Appendixes D and E (PKI Management Message Profiles), with the following exceptions:

- signature-based protection is the default protection; an initial PKI management operation may also use HMAC,

- certification of a second key pair within the same PKI management operation is not supported,

- proof-of-possession (POPO) with self-signature of the certTemplate according to [RFC4211] section 4.1 clause 3 is the recommended default POPO method (deviations are possible by EEs when requesting central key generation and by (L)RAs when using raVerified),

- confirmation of newly enrolled certificates may be omitted, and

- all PKI management operations consist of request-response message pairs originating at the EE, i.e., announcement messages are omitted.

The profile specified in this document is compatible with the CMP profile for UMTS, LTE, and 5G network domain security and authentication framework [ETSI-3GPP], except that:

- protection of initial PKI management operations may be HMAC-based,

- the subject name is mandatory in certificate templates, and

- confirmation of newly enrolled certificates may be omitted.

The profile specified in this document is compatible with the CMP profile for on-line key management in rail networks as specified in UNISIG subset-137 [UNISIG], except that:

- As stated in Section 4.1.1 a CMP message SHALL only consist of one certificate request (CertReqMsg). Therefore, UNISIG is in conflict with this document as subset-137 allows to transport more than one certificate request.

- as of RFC 4210 [RFC4210] the messageTime is required to be Greenwich Mean Time coded as generalizedTime (Note: While UNISIG explicitly states that the messageTime in required to be 'UTC time', it is not clear if this means a coding as UTCTime or generalizedTime and if other time zones than Greenwich Mean Time shall be allowed. Therefore, UNISG may be in conflict with RFC 4210 [RFC4210]. Both time formats are described in RFC 5280 [RFC5280] section 4.1.2.5.), and

- in case the request message is MAC protected, also the response, certConf, and pkiConf messages have a MAC-based protection (Note: if changing to signature protection of the response the caPubs field cannot be used securely anymore.).

1.5. Scope of this document

This document specifies requirements on generating PKI management messages on the sender side. It does not specify strictness of verification on the receiving side and how in detail to handle error cases.

Especially on the EE side this profile aims at a lightweight protocol that can be implemented on more constrained devices. On the side of the central PKI management entities the profile accepts higher resources needed.

For the sake of robustness and preservation of security properties implementations should, as far as security is not affected, adhere to Postel's law: "Be conservative in what you do, be liberal in what you accept from others" (often reworded as: "Be conservative in what you send, be liberal in what you accept").

When in Section 3, Section 4, and Section 5 a field of the ASN.1 syntax as defined in RFC 4210 [RFC4210] and RFC 4211 [RFC4211] is not explicitly specified, it SHOULD not be used by the sending entity. The receiving entity MUST NOT require its absence and if present MUST gracefully handle its presence.

1.6. Structure of this document

Section 2 introduces the general PKI architecture and approach to certificate management using CMP that is assumed in this document. Then it enlists the PKI management operations specified in this document and describes them in general words. The list of supported PKI management operations is divided into mandatory, recommended, and optional ones.

Section 3 profiles the CMP message header, protection, and extraCerts section as they are general elements of CMP messages.

Section 4 profiles the exchange of CMP messages between an EE and the first PKI management entities. There are various flavors of certificate enrollment requests optionally with polling, revocation, error handling, and general support PKI management operations.

Section 5 profiles the exchange between PKI management entities. These are in the first place the forwarding of messages coming from or going to an EE. This includes also initiating delayed delivery of messages, which involves polling. Additionally, it specifies PKI management operations where a PKI management entity manages certificates on behalf of an EE or for itself.

Section 6 outlines different mechanisms for CMP message transfer, namely http-based transfer as already specified in [RFC6712], using an additional TLS layer, or offline file-based transport. CoAP [RFC7252] and piggybacking CMP messages on other protocols is out of scope and left for further documents.

1.7. Convention and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case use of these words are not to be interpreted as carrying significance described in RFC 2119.

Technical terminology is used in conformance with RFC 4210 [RFC4210], RFC 4211 [RFC4211], RFC 5280 [RFC5280], and IEEE 802.1AR [IEEE802.1AR]. The following key words are used:

CA: Certification authority, which issues certificates.

RA: Registration authority, an optional system component to which a CA delegates certificate management functions such as authorization checks.

LRA: Local registration authority, an optional RA system component with proximity to the end entities.

KGA: Key generation authority, an optional system component, typically co-located with an LRA, RA, or CA, that offers key generation services to end entities.

EE: End entity, a user, device, or service that holds a PKI certificate. An identifier for the EE is given as the subject of its certificate.

The following terminology is reused from RFC 4210 [RFC4210] and used as follows:

PKI management operation: All CMP messages belonging to one transaction context. The transaction is identified in the transactionID field of the message header.

PKI management entity: All central PKI entities like LRA, RA and CA.

PKI entity: EEs and PKI management entities

2. Architecture and use cases

2.1. Solution architecture

Typically, a machine EE will be equipped with a manufacturer issued certificate during production. Such a manufacturer issued certificate is installed during production to identify the device throughout its lifetime. This manufacturer certificate can be used to protect the initial enrollment of operational certificates after installation of the EE in a plant or industrial network. An operational certificate is issued by the owner or operator of the device to identify the device during operation, e.g., within a security protocol like IPSec, TLS, or SSH. In IEEE 802.1AR [IEEE802.1AR] a manufacturer certificate is called IDevID certificate and an operational certificate is called LDevID certificate.

All certificate management transactions specified in this document are initiated by the EE. The EE creates a CMP request message, protects it using its manufacturer or operational certificate, if available, and sends it to its locally reachable PKI component. This PKI component may be an LRA, RA, or the CA, which checks the request, responds to it itself, or forwards the request upstream to the next PKI component. In case an (L)RA changes the CMP request message header or body or wants to prove a successful verification or authorization, it can apply a protection of its own. Especially the communication between an LRA and RA can be performed synchronously or asynchronously. Synchronous communication describes a timely uninterrupted communication between two communication partners, while asynchronous communication is not performed in a timely consistent manner, e.g., because of a delayed message delivery.

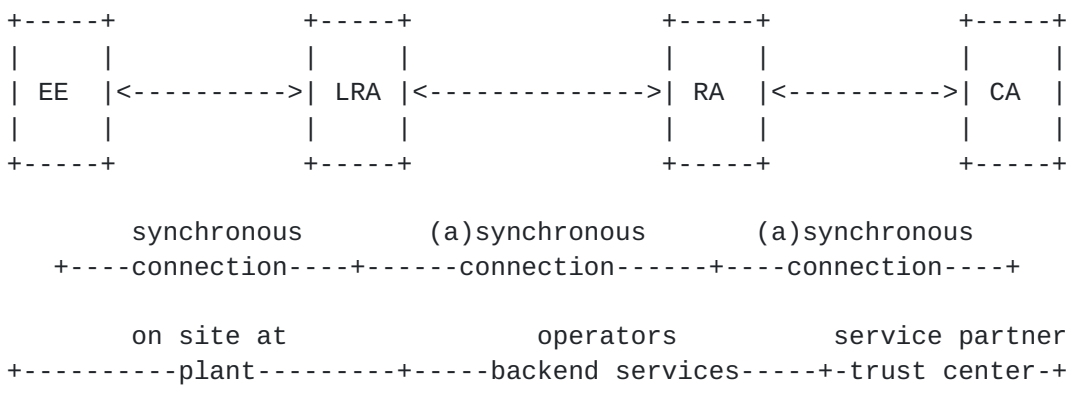

Figure 1: Certificate management on site

In operation environments a layered LRA-RA-CA architecture can be deployed, e.g., with LRAs bundling requests from multiple EEs at dedicated locations and one (or more than one) central RA aggregating the requests from multiple LRAs. Every (L)RA in this scenario will have its own dedicated certificate containing an extended key usage as specified in CMP Updates [I-D.ietf-lamps-cmp-updates] and private key allowing it to protect CMP messages it processes (CMP signing key/certificate). The figure above shows an architecture using one LRA and one RA. It is also possible to have only an RA or multiple LRAs and/or RAs. Depending on the network infrastructure, the communication between different PKI management entities may be synchronous online communication, delayed asynchronous communication, or even offline file transfer.

This profile focusses on specifying the pull model, where the EE always requests a specific PKI management operation. CMP response messages, especially in case of central key generation, as described in Section 4.1.6, could also be used proactively to implement the push model towards the EE.

Third-party CAs typically implement different variants of CMP or even use proprietary interfaces for certificate management. Therefore, the LRA or the RA may need to adapt the exchanged CMP messages to the flavor of communication required by the CA.

2.2. Basic generic CMP message content

Section 3 specifies the generic parts of the CMP messages as used later in Section 4 and Section 5.

- Header of a CMP message; see Section 3.1.
- Protection of a CMP message; see Section 3.2.
- ExtraCerts field of a CMP message; see Section 3.3.

2.3. Supported PKI management operations

Following the outlined scope from Section 1.5, this section gives a brief overview of the PKI management operations specified in Section 4 and Section 5 and points out whether an implementation by compliant EE or PKI management entities is mandatory, recommended or optional.

2.3.1. Mandatory PKI management operations

The mandatory PKI management operations in this document shall limit the overhead of certificate management for more constrained devices to the most crucial types of operations.

Section 4 - End Entity focused PKI management operations

- Request a certificate from a new PKI with signature protection; see Section 4.1.1.
- Request to update an existing certificate with signature protection; see Section 4.1.3.
- Error reporting; see Section 4.3.

Section 5 - LRA and RA focused PKI management operations

- Forward messages without changes; see Section 5.1.1.
- Forward messages with replaced protection and keeping the original proof-of-possession; see Section 5.1.2.1.
- Forward messages with replaced protection and raVerified as proof-of-possession; see Section 5.1.2.2.
- Error reporting; see Section 5.3.

2.3.2. Recommended PKI management operations

Additional recommended PKI management operations shall support some more complex scenarios, that are considered as beneficial for environments with more specific boundary conditions.

Section 4 - End Entity focused PKI management operations

- Request a certificate from a PKI with MAC protection; see Section 4.1.4.
- Revoke an own certificate.

Section 5 - LRA and RA focused PKI management operations

- Revoke another's entities certificate.

2.3.3. Optional PKI management operations

The optional PKI management operations support specific requirements seen only in a subset of environments.

Section 4 - End Entity focused PKI management operations

- Request a certificate from a trusted PKI with signature protection; see Section 4.1.2.
- Request a certificate from a legacy PKI using a PKCS#10 [RFC2986] request; see Section 4.1.5.
- Add central generation of a key pair to a certificate request; see Section 4.1.6. If central key generation is supported, the key agreement key management technique is REQUIRED to be supported, and the key transport and symmetric key-encryption key management techniques are OPTIONAL.
- Handle delayed enrollment due to asynchronous message delivery; see Section 4.1.7.
- Additional support messages, e.g., to update a root CA certificate or to request an RFC 8366 [RFC8366] voucher; see Section 4.4.

Section 5 - LRA and RA focused PKI management operations

- Forward messages with additional protection; see Section 5.1.3
- Initiate delayed enrollment due to asynchronous message delivery; see Section 5.1.4.

2.4. CMP message transport

On different links between PKI entities, e.g., EE<->RA and RA<->CA, different transport MAY be used. As CMP has only very limited requirement regarding the mechanisms used for message transport and in different environments different transport mechanisms are supported, e.g. HTTP, CoAP, or even offline files based, this document requires no specific transport protocol to be supported by all conforming implementations.

HTTP transfer is RECOMMENDED to use for all PKI entities, but there is no transport specified as mandatory to be flexible for devices with special constraints to choose whatever transport is suitable.

Recommended transport

- Transfer CMP messages using HTTP; see Section 6.2.

Optional transport

- Transfer CMP messages using HTTPS with certificate-based authentication; see Section 6.3.
- Transfer CMP messages using HTTPS with shared-secret based protection; see Section 6.4.
- File-based CMP message transport.

3. Generic parts of the PKI message

To reduce redundancy in the text and to ease implementation, the contents of the header, protection, and extraCerts fields of the CMP messages used in the transactions specified in Section 4 and Section 5 are standardized to the maximum extent possible. Therefore, the generic parts of a CMP message are described centrally in this section.

As described in section 5.1 of [RFC4210], all CMP messages have the following general structure:

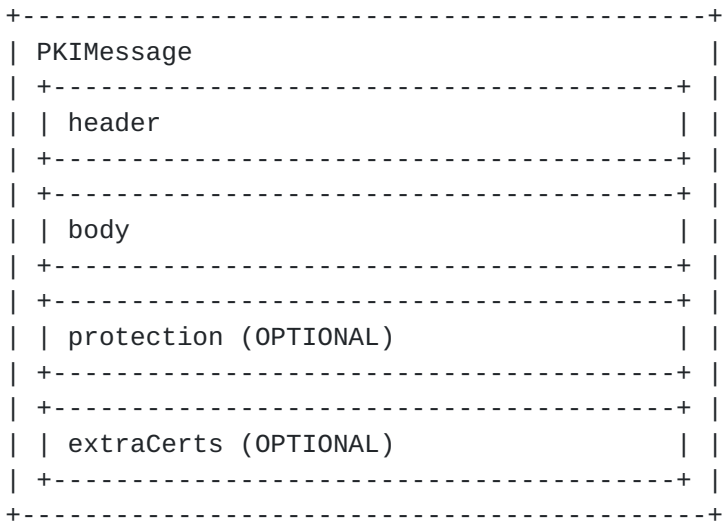

Figure 2: CMP message structure

The general contents of the message header, protection, and extraCerts fields are specified in the Section 3.1 to Section 3.3.

In case a specific CMP message needs different contents in the header, protection, or extraCerts fields, the differences are described in the respective message.

The CMP message body contains the message-specific information. It is described in the context of Section 4 and Section 5.

The behavior in case an error occurs while handling a CMP message is described in Section 5.3.

3.1. General description of the CMP message header

This section describes the generic header field of all CMP messages with signature-based protection. The only variations described here are in the fields recipient, transactionID, and recipNonce of the first message of a PKI management operation.

In case a message has MAC-based protection the changes are described in the respective section. The variations will affect the fields sender, protectionAlg, and senderKID.

For requirements about proper random number generation please refer to [RFC4086]. Any message-specific fields or variations are described in the respective sections of this chapter.

```
header
  pvno                     REQUIRED
    -- MUST be set to 2 to indicate CMP V2
  sender                   REQUIRED
    -- MUST be the subject of the protection certificate used for,
    -- the certificate for the private key used to sign the message
  recipient                REQUIRED
    -- SHOULD be the name of the intended recipient and
    -- MAY be a NULL_DN if the sender does not know the DN of
    -- the recipient
    -- If this is the first message of a transaction: SHOULD be the
    -- subject of the issuing CA certificate
    -- In all other messages: SHOULD be the same name as in the
    -- sender field of the previous message in this transaction
  messageTime              RECOMMENDED
    -- MUST be the time at which the message was produced, if
    -- present
  protectionAlg            REQUIRED
    -- MUST be the algorithm identifier of the signature algorithm or
    -- id-PasswordBasedMac algorithm used for calculation of the
    -- protection bits
    -- The signature algorithm MUST be consistent with the
    -- subjectPublicKeyInfo field of the signer's certificate
    -- The hash algorithm used SHOULD be SHA-256
    algorithm              REQUIRED
    -- MUST be the OID of the signature algorithm, like
    -- sha256WithRSAEncryption or ecdsa-with-SHA256, or
```

```
      -- id-PasswordBasedMac
    senderKID                   RECOMMENDED
      -- MUST be the SubjectKeyIdentifier, if available, of the
      -- protection certificate
    transactionID               REQUIRED
      -- If this is the first message of a transaction:
      -- MUST be 128 bits of random data for the start of a
      -- transaction to reduce the probability of having the
      -- transactionID already in use at the server
      -- In all other messages:
      -- MUST be the value from the previous message in the same
      -- transaction
    senderNonce                 REQUIRED
      -- MUST be fresh 128 random bits
    recipNonce                  RECOMMENDED
      -- If this is the first message of a transaction: SHOULD be
      -- absent
      -- In all other messages: MUST be present and contain the value
      -- from senderNonce of the previous message in the same
      -- transaction
    generalInfo                 OPTIONAL
      implicitConfirm           OPTIONAL
      -- The field is optional though it only applies to
      -- ir/cr/kur/p10cr requests and ip/cp/kup response messages
      -- Add to request messages to request omit sending certConf
      -- message
      -- Add to response messages to confirm omit sending certConf
      -- message
        ImplicitConfirmValue    REQUIRED
      -- ImplicitConfirmValue of the request message MUST be NULL if
      -- the EE wants to request not to send a confirmation message
      -- ImplicitConfirmValue MUST be set to NULL if the (L)RA/CA
      -- wants to grant not sending a confirmation message
```

3.2. General description of the CMP message protection

This section describes the generic protection field of all CMP messages with signature-based protection. The certificate for the private key used to sign a CMP message is called 'protection certificate'.

```
protection                  RECOMMENDED
    -- MUST contain the signature calculated using the signature
    -- algorithm specified in protectionAlg
```

Generally, CMP message protection is required for CMP messages, but there are cases where protection of error messages as specified in

Section 4.3 and Section 5.3 is not possible and therefore MAY be omitted.

For MAC-based protection as specified in Section 4.1.4 major differences apply as described in the respective section.

The CMP message protection provides, if available, message origin authentication and integrity protection for the CMP message header and body. The CMP message extraCerts is not covered by this protection.

NOTE: The extended key usages specified in CMP Updates [I-D.ietf-lamps-cmp-updates] can be used for authorization of a sending PKI management entity.

NOTE: The requirements for checking certificates given in [RFC5280] MUST be followed for the CMP message protection. In case the CMP signer certificate is not the CA certificate that signed the newly issued certificate, certificate status checking SHOULD be used for the CMP signer certificates of communication partners.

3.3. General description of CMP message extraCerts

This section describes the generic extraCerts field of all CMP messages with signature-based protection. If extraCerts are required, recommended, or optional is specified in the respective PKI management operation.

```
extraCerts
    -- SHOULD contain the protection certificate together with its
    -- chain, if needed and the self-signed root certificate SHOULD
    -- be omitted
    -- If present, the first certificate in this field MUST
    -- be the protection certificate and each following certificate
    -- SHOULD directly certify the one immediately preceding it.
    -- Self-signed certificates SHOULD be omitted from extraCerts
    -- and MUST NOT be trusted based on the listing in extraCerts
    -- in any case
```

Note: For maximum compatibility, all implementations SHOULD be prepared to handle potentially additional and arbitrary orderings of the certificates, except that the protection certificate is the first certificate in extraCerts.

4. End Entity focused PKI management operations

This chapter focuses on the communication of the EE and the first PKI management entities it talks to. Depending on the network and PKI solution, this will either be the LRA, the RA or the CA.

Profiles of the Certificate Management Protocol (CMP) [RFC4210] handled in this section cover the following PKI management operations:

- Requesting a certificate from a PKI with variations like initial requests and updating, central key generation and different protection means
- Revocation of a certificate
- General messages for further support functions

These operations mainly specify the message body of the CMP messages and utilize the specification of the message header, protection and extraCerts as specified in Section 4.

The behavior in case an error occurs is described in Section 4.3.

This chapter is aligned to Appendix D and Appendix E of [RFC4210]. The general rules for interpretation stated in Appendix D.1 in [RFC4210] need to be applied here, too.

This document does not mandate any specific supported algorithms like Appendix D.2 of [RFC4210], [ETSI-3GPP], and [UNISIG] do. Using the message sequences described here require agreement upon the algorithms to support and thus the algorithm identifiers for the specific target environment.

4.1. Requesting a new certificate from a PKI

There are different approaches to request a certificate from a PKI.

These approaches differ on the one hand in the way the EE can authenticate itself to the PKI it wishes to get a new certificate from and on the other hand in its capabilities to generate a proper new key pair. The authentication means may be as follows:

- Using a certificate from a trusted PKI and the corresponding private key, e.g., a manufacturer issued certificate
- Using the certificate to be updated and the corresponding private key

o Using a shared secret known to the EE and the PKI

Typically, such EE requests a certificate from a CA. When the PKI management entity responds with a message containing a certificate, the EE MUST reply with a confirmation message. The PKI management entity then MUST send confirmation back, closing the transaction.

The message sequences in this section allow the EE to request certification of a locally generated public-private key pair. For requirements about proper random number and key generation please refer to [RFC4086]. The EE MUST provide a signature-based proof-of-possession of the private key associated with the public key contained in the certificate request as defined by [RFC4211] section 4.1 case 3. To this end it is assumed that the private key can technically be used as signing key. The most commonly used algorithms are RSA and ECDSA, which can technically be used for signature calculation regardless of potentially intended restrictions of the key usage.

The requesting EE provides the binding of the proof-of-possession to its identity by signature-based or MAC-based protection of the CMP request message containing that POPO. The PKI management entity needs to verify whether this EE is authorized to obtain a certificate with the requested subject and other fields and extensions. Especially when removing the protection provided by the EE and applying a new protection, the PKI management entity MUST verify in particular the included proof-of-possession self-signature of the certTemplate using the public key of the requested certificate and MUST check that the EE, as authenticated by the message protection, is authorized to request a certificate with the subject as specified in the certTemplate (see Section 5.1.2).

There are several ways to install the Root CA certificate of a new PKI on an EE. The installation can be performed in an out-of-band manner, using general messages, a voucher [RFC8366], or other formats for enrollment, or in-band of CMP by the caPubs field in the certificate response message. In case the installation of the new root CA certificate is performed using the caPubs field, the certificate response message MUST be properly authenticated, and the sender of this message MUST be authorized to install new root CA certificates on the EE. This authorization can be indicated by using pre-shared keys for the CMP message protection.

4.1.1. Request a certificate from a new PKI with signature protection

This PKI management operation should be used by an EE to request a certificate of a new PKI using an existing certificate from an external PKI, e.g., a manufacturer certificate, to prove its identity

to the new PKI. The EE already has established trust in this new PKI it is about to enroll to, e.g., by voucher exchange or configuration means. The initialization request message is signature-protected using the existing certificate.

Preconditions:

1 The EE MUST have a certificate enrolled by an external PKI in advance to this PKI management operation to authenticate itself to the PKI management entity using signature-based protection, e.g., using a manufacturer issued certificate.

2 The EE SHOULD know the subject name of the new CA it requests a certificate from; this name MAY be established using an enrollment voucher, the issuer field from a the CertReqTemplate response message, or other configuration means. If the EE does not know the name of the CA, the PKI management entity MUST know where to route this request to.

3 The EE MUST authenticate responses from the PKI management entity; trust MAY be established using an enrollment voucher or other configuration means.

4 The PKI management entity MUST trust the external PKI the EE uses to authenticate itself; trust MAY be established using some configuration means.

This PKI management operation is like that given in [RFC4210] Appendix E.7.

Message flow:

Step#	EE				PKI management entity
1	format ir				
2		->	ir	->	
3					handle, re-protect or forward ir
4					format or receive ip
5					possibly grant implicit confirm
6		<-	ip	<-	
7	handle ip				
8					In case of status "rejection" in the ip message, no certConf and pkiConf are sent
9	format certConf (optional)				
10		->	certConf	->	
11					handle, re-protect or forward certConf
12					format or receive pkiConf
13		<-	pkiconf	<-	
14	handle pkiConf (optional)				

For this PKI management operation, the EE MUST include exactly one single CertReqMsg in the ir. If more certificates are required, further requests MUST be sent using separate CMP messages. If the EE wants to omit sending a certificate confirmation message after receiving the ip to reduce the number of protocol messages exchanged in this PKI management operation, it MUST request this by including the implicitConfirm extension in the ir.

If the CA accepts the certificate request it MUST return the new certificate in the certifiedKeyPair field of the ip message. If the EE requested to omit sending a certConf message after receiving the ip, the PKI management entity MAY confirm it by also including the implicitConfirm extension or MAY rejects it by omitting the implicitConfirm field in the ip.

If the EE did not request implicit confirmation or the request was not granted by the PKI management entity the confirmation as follows MUST be performed. If the EE successfully receives the certificate and accepts it, the EE MUST send a certConf message, which MUST be answered by the PKI management entity with a pkiConf message. If the PKI management entity does not receive the expected certConf message in time it MUST handle this like a rejection by the EE.

If the certificate request was refused by the CA, the PKI management entity must return an ip message containing the status code "rejection" and no certifiedKeyPair field. Such an ip message MUST NOT be followed by the certConf and pkiConf messages.

Detailed message description:

Certification Request -- ir

```
Field                         Value

header
    -- As described in section 3.1

body
    -- The request of the EE for a new certificate
  ir                          REQUIRED
    -- MUST be exactly one CertReqMsg
    -- If more certificates are required, further requests MUST be
    -- packaged in separate PKI Messages
    certReq                   REQUIRED
      certReqId               REQUIRED
    -- MUST be set to 0
      certTemplate            REQUIRED
        version               OPTIONAL
    -- MUST be 2 if supplied.
        subject               REQUIRED
    -- MUST contain the suggested subject name of the EE
    -- certificate
        publicKey             REQUIRED
          algorithm           REQUIRED
    -- MUST include the subject public key algorithm ID and value
    -- In case a central key generation is requested, this field
    -- contains the algorithm and parameter preferences of the
    -- requesting entity regarding the to-be-generated key pair
          subjectPublicKey    REQUIRED
    -- MUST contain the public key to be included into the requested
    -- certificate in case of local key-generation
    -- MUST contain a zero-length BIT STRING in case a central key
    -- generation is requested
    -- MUST include the subject public key algorithm ID and value
        extensions            OPTIONAL
    -- MAY include end-entity-specific X.509 extensions of the
    -- requested certificate like subject alternative name,
    -- key usage, and extended key usage
    Popo                      REQUIRED
      POPOSigningKey          OPTIONAL
    -- MUST be used in case subjectPublicKey contains a public key
```

```
      -- MUST be absent in case subjectPublicKey contains a
      -- zero-length BIT STRING
          poposkInput           PROHIBITED
      -- MUST NOT be used because subject and publicKey are both
      -- present in the certTemplate
          algorithmIdentifier   REQUIRED
      -- The signature algorithm MUST be consistent with the
      -- publicKey field of the certTemplate
      -- The hash algorithm used SHOULD be SHA-256
          signature             REQUIRED
      -- MUST be the signature computed over the DER-encoded
      -- certTemplate

  protection                    REQUIRED
      -- As described in section 3.2

  extraCerts                    REQUIRED
      -- As described in section 3.3


  Certification Response -- ip

  Field                         Value

  header
      -- As described in section 3.1

  body
      -- The response of the CA to the request as appropriate
    ip                          REQUIRED
      caPubs                    OPTIONAL
      -- MAY be used
      -- If used it MUST contain only the root certificate of the
      -- certificate contained in certOrEncCert
      response                  REQUIRED
      -- MUST be exactly one CertResponse
        certReqId               REQUIRED
      -- MUST be set to 0
        status                  REQUIRED
      -- PKIStatusInfo structure MUST be present
          status                REQUIRED
      -- positive values allowed: "accepted", "grantedWithMods"
      -- negative values allowed: "rejection"
      -- In case of rejection certConf and pkiConf messages MUST NOT
      -- be sent
          statusString          OPTIONAL
      -- MAY be any human-readable text for debugging, logging or to
      -- display in a GUI
```

```
          failInfo              OPTIONAL
       -- MUST be present if status is "rejection" and in this case
       -- the transaction MUST be terminated
       -- MUST be absent if the status is "accepted" or
       -- "grantedWithMods"
         certifiedKeyPair       OPTIONAL
       -- MUST be present if status is "accepted" or "grantedWithMods"
       -- MUST be absent if status is "rejection"
          certOrEncCert         REQUIRED
       -- MUST be present when certifiedKeyPair is present
            certificate         REQUIRED
       -- MUST be present when certifiedKeyPair is present
       -- MUST contain the newly enrolled X.509 certificate
          privateKey            OPTIONAL
       -- MUST be absent in case of local key-generation
       -- MUST contain the encrypted private key in an EnvelopedData
       -- structure as specified in section 5.1.5 in case the private
       -- key was generated centrally

   protection                   REQUIRED
       -- As described in section 3.2

   extraCerts                   REQUIRED
       -- As described in section 3.3
       -- MUST contain the chain of the certificate present in
       -- certOrEncCert, the self-signed root certificate SHOULD be
       -- omitted
       -- Duplicate certificates MAY be omitted


   Certificate Confirmation -- certConf

   Field                        Value

   header
       -- As described in section 3.1

   body
       -- The message of the EE sends confirmation to the PKI
       -- management entity to accept or reject the issued certificates
     certConf                   REQUIRED
       -- MUST be exactly one CertStatus
       CertStatus               REQUIRED
         certHash               REQUIRED
       -- MUST be the hash of the certificate, using the same hash
       -- algorithm as used to create the certificate signature
         certReqId              REQUIRED
       -- MUST be set to 0
```

```
        status                  RECOMMENDED
      -- PKIStatusInfo structure SHOULD be present
      -- Omission indicates acceptance of the indicated certificate
         status                 REQUIRED
      -- positive values allowed: "accepted"
      -- negative values allowed: "rejection"
         statusString           OPTIONAL
      -- MAY be any human-readable text for debugging, logging, or to
      -- display in a GUI
         failInfo               OPTIONAL
      -- MUST be present if status is "rejection"
      -- MUST be absent if the status is "accepted"

  protection                    REQUIRED
      -- As described in section 3.2
      -- MUST use the same certificate as for protection of the ir

  extraCerts                    RECOMMENDED
      -- SHOULD contain the protection certificate together with its
      -- chain, but MAY be omitted if the message size is critical and
      -- the PKI management entity did cash the extraCerts from the ir
      -- If present, the first certificate in this field MUST be the
      -- certificate used for signing this message
      -- Self-signed certificates SHOULD NOT be included in
      -- extraCerts and
      -- MUST NOT be trusted based on the listing in extraCerts in
      -- any case


  PKI Confirmation -- pkiconf

  Field                         Value

  header
      -- As described in section 3.1

  body
    pkiconf                     REQUIRED
      -- The content of this field MUST be NULL

  protection                    REQUIRED
      -- As described in section 3.2
      -- SHOULD use the same certificate as for protection of the ip

  extraCerts                    RECOMMENDED
      -- SHOULD contain the protection certificate together with its
      -- chain, but MAY be omitted if the message size is critical and
      -- the PKI management entity did cash the extraCerts from the ip
```

```
      -- If present, the first certificate in this field MUST be the
      -- certificate used for signing this message
      -- Self-signed certificates SHOULD NOT be included in extraCerts
      -- and
      -- MUST NOT be trusted based on the listing in extraCerts in
      -- any case
```

4.1.2. Request a certificate from a trusted PKI with signature protection

This PKI management operation should be used by an EE to request an additional certificate of the same PKI it already has certificates from. The EE uses one of these existing certificates to prove its identity. The certificate request message is signature-protected using this certificate.

The general message flow for this PKI management operation is the same as given in Section 4.1.1.

Preconditions:

1 The EE MUST have a certificate enrolled by the PKI it requests another certificate from in advance to this PKI management operation to authenticate itself to the PKI management entity using signature-based protection.

2 The EE SHOULD know the subject name of the CA it requests a certificate from; this name MAY be established using an enrollment voucher, the issuer field from a the CertReqTemplate response message, or other configuration means. If the EE does not know the name of the CA, the PKI management entity MUST know where to route this request to.

3 The EE MUST authenticate responses from the PKI management entity; trust MUST be established using an enrollment voucher or other configuration means.

4 The PKI management entity MUST trust the current PKI; trust MAY be established using some configuration means.

The message sequence for this PKI management operation is like that given in [RFC4210] Appendix D.5.

The message sequence for this PKI management operation is identical to that given in Section 4.1.1, with the following changes:

1 The body of the first request and response MUST be cr and cp, respectively.

2 The caPubs field in the cp message SHOULD be absent.

4.1.3. Update an existing certificate with signature protection

This PKI management operation should be used by an EE to request an update of one of the certificates it already has and that is still valid. The EE uses the certificate it wishes to update to prove its identity and possession of the private key for the certificate to be updated to the PKI. Therefore, the key update request message is signed using the certificate that is to be updated.

The general message flow for this PKI management operation is the same as given in Section 4.1.1.

Preconditions:

1 The certificate the EE wishes to update MUST NOT be expired or revoked.

2 A new public-private key pair SHOULD be used.

The message sequence for this PKI management operation is like that given in [RFC4210] Appendix D.6.

The message sequence for this PKI management operation is identical to that given in Section 4.1.1, with the following changes:

1 The body of the first request and response MUST be kur and kup, respectively.

2 Protection of the kur MUST be performed using the certificate to be updated.

3 The subject field of the CertTemplate MUST contain the subject name of the existing certificate to be updated, without modifications.

4 The CertTemplate MUST contain the subject, issuer and publicKey fields only.

5 The oldCertId control SHOULD be used to make clear, even in case an (L)RA changes the message protection, which certificate is to be updated.

6 The caPubs field in the kup message MUST be absent.

As part of the certReq structure of the kur the control is added right after the certTemplate.

```
    controls
      type                    RECOMMENDED
    -- MUST be the value id-regCtrl-oldCertID, if present
      value
        issuer                REQUIRED
        serialNumber          REQUIRED
    -- MUST contain the issuer and serialNumber of the certificate
    -- to be updated
```

4.1.4. Request a certificate from a PKI with MAC protection

This PKI management operation should be used by an EE to request a certificate of a new PKI without having a certificate to prove its identity to the target PKI, but there is a shared secret established between the EE and the PKI. Therefore, the initialization request is MAC-protected using this shared secret. The PKI management entity checking the MAC-protection SHOULD replace this protection according to Section 5.1.2 in case the next hop does not know the shared secret.

For requirements with regard to proper random number and key generation please refer to [RFC4086].

The general message flow for this PKI management operation is the same as given in Section 4.1.1.

Preconditions:

1 The EE and the PKI management entity MUST share a symmetric key, this MAY be established by a service technician during initial local configuration.

2 The EE SHOULD know the subject name of the new CA it requests a certificate from; this name MAY be established using an enrollment voucher, the issuer field from a the CertReqTemplate response message, or other configuration means. If the EE does not know the name of the CA, the PKI management entity MUST know where to route this request to.

3 The EE MUST authenticate responses from the PKI management entity; trust MAY be established using the shared symmetric key.

The message sequence for this PKI management operation is like that given in [RFC4210] Appendix D.4.

The message sequence for this PKI management operation is identical to that given in Section 4.1.1, with the following changes:

1 The protection of all messages MUST be calculated using Message Authentication Code (MAC); the protectionAlg field MUST be id-PasswordBasedMac as described in section 5.1.3.1 of [RFC4210].

2 The sender MUST contain a name representing the originator of the message. The senderKID MUST contain a reference all participating entities can use to identify the symmetric key used for the protection, e.g., the username of the EE.

3 The extraCerts of the ir, certConf, and pkiConf messages MUST be absent.

4 The extraCerts of the ip message MUST contain the chain of the issued certificate and root certificates SHOULD not be included and MUST NOT be directly trusted in any case.

Part of the protectionAlg structure, where the algorithm identifier MUST be id-PasswordBasedMac, is a PBMParameter sequence. The fields of PBMParameter SHOULD remain constant for message protection throughout this PKI management operation to reduce the computational overhead.

```
    PBMParameter              REQUIRED
      salt                    REQUIRED
    -- MUST be the random value to salt the secret key
      owf                     REQUIRED
    -- MUST be the algorithm identifier for the one-way function
    -- used
    -- The one-way function SHA-1 MUST be supported due to
    -- [RFC4211] requirements, but SHOULD NOT be used any more
    -- SHA-256 SHOULD be used instead
      iterationCount          REQUIRED
    -- MUST be a limited number of times the one-way function is
    -- applied
    -- To prevent brute force and dictionary attacks a reasonable
    -- high number SHOULD be used
      mac                     REQUIRED
    -- MUST be the algorithm identifier of the MAC algorithm used
    -- The MAC function HMAC-SHA1 MUST be supported due to
    -- [RFC4211] requirements, but SHOULD NOT be used any more
    -- HMAC-SHA-256 SHOULD be used instead
```

4.1.5. Request a certificate from a legacy PKI using PKCS#10 request

This PKI management operation should be used by an EE to request a certificate of a legacy PKI only capable to process PKCS#10 [RFC2986] certification requests. The EE can prove its identity to the target PKI by using various protection means as described in Section 4.1.1 or Section 4.1.4.

In contrast to the other PKI management operations described in Section 4.1, this transaction uses PKCS#10 [RFC2986] instead of CRMF [RFC4211] for the certificate request for compatibility reasons with legacy CA systems that require a PKCS#10 certificate request and cannot process CRMF [RFC4211] requests. In such case the PKI management entity MUST extract the PKCS#10 certificate request from the p10cr and provides it separately to the CA.

The general message flow for this PKI management operation is the same as given in Section 4.1.1, but the public key is contained in the subjectPKInfo of the PKCS#10 certificate request.

Preconditions:

1 The EE MUST either have a certificate enrolled from this or any other accepted PKI, or a shared secret known to the PKI and the EE to authenticate itself to the RA.

2 The EE SHOULD know the subject name of the CA it requests a certificate from; this name MAY be established using an enrollment voucher, the issuer field from a the CertReqTemplate response message, or other configuration means. If the EE does not know the name of the CA, the RA MUST know where to route this request to.

3 The EE MUST authenticate responses from the RA; trust MAY be established by an available root certificate, using an enrollment voucher, or other configuration means.

4 The RA MUST trust the current or the PKI the EE uses to authenticate itself; trust MAY be established by a corresponding available root certificate or using some configuration means.

The message sequence for this PKI management operation is identical to that given in Section 4.1.1, with the following changes:

1 The body of the first request and response MUST be p10cr and cp, respectively.

2 The certReqId in the cp message MUST be 0.

3 The caPubs field in the cp message SHOULD be absent.

Detailed description of the p10cr message:

Certification Request -- p10cr

```
Field                          Value

header
    -- As described in section 3.1

body
    -- The request of the EE for a new certificate using a PKCS#10
    -- certificate request
  p10cr                        REQUIRED
    certificationRequestInfo   REQUIRED
      version                  REQUIRED
    -- MUST be set to 0 to indicate PKCS#10 V1.7
      subject                  REQUIRED
    -- MUST contain the suggested subject name of the EE
      subjectPKInfo            REQUIRED
        algorithm              REQUIRED
    -- MUST include the subject public key algorithm ID
        subjectPublicKey       REQUIRED
    -- MUST include the subject public key algorithm value
      attributes               OPTIONAL
    -- MAY contain a set of end-entity-specific fields or X.509
    -- extensions to be included in the requested certificate or used
    -- otherwise
    signatureAlgorithm         REQUIRED
    -- The signature algorithm MUST be consistent with the
    -- subjectPKInfo field. The hash algorithm used SHOULD be SHA-256
    signature                  REQUIRED
    -- MUST containing the self-signature for proof-of-possession

protection                     REQUIRED
    -- As described in section 3.2

extraCerts                     REQUIRED
    -- As described in section 3.3
```

4.1.6. Generate the key pair centrally at the PKI management entity

This functional extension can be applied in combination with certificate enrollment as described in Section 4.1.1 and Section 4.1.4. The functional extension can be used in case an EE is not able or is not willing to generate its new public-private key pair itself. It is a matter of the local implementation which PKI

management entity will perform the key generation. This entity MUST have a certificate containing the additional extended key usage extension id-kp-cmcKGA to be identified by the EE as a legitimate key-generation authority. In case the PKI management entity generated the new key pair for the EE, it can use Section 4.1.1 to Section 4.1.4 to request the certificate for this key pair as usual.

Generally speaking, in a machine-to-machine scenario it is strongly preferable to generate public-private key pairs locally at the EE. Together with proof-of-possession of the private key in the certification request, this is to make sure that only the entity identified in the newly issued certificate is the only entity who ever hold the private key.

There are some cases where an EE is not able or not willing to locally generate the new key pair. Reasons for this may be the following:

o Lack of sufficient initial entropy.

Note: Good random numbers are not only needed for key generation, but also for session keys and nonces in any security protocol. Therefore, we believe that a decent security architecture should anyways support good random number generation on the EE side or provide enough entropy for the RNG seed during manufacturing to guarantee good initial pseudo-random number generation.

o Due to lack of computational resources, e.g., in case of RSA keys.

Note: As key generation can be performed in advance to the certificate enrollment communication, it is typical not time critical.

Note: Besides the initial enrollment right after the very first bootup of the device, where entropy available on the device may be insufficient, we do not see any good reason for central key generation.

Note: As mentioned in Section 2.1 central key generation may be required in a push model, where the certificate response message is transferred by the PKI management entity to the EE without receiving a previous request message.

If the EE wishes to request central key generation, it MUST fill the subjectPublicKey field in the certTemplate structure of the request message with a zero-length BIT STRING. This indicates to the PKI management entity that a new key pair shall be generated centrally on behalf of the EE.

Note: As the protection of centrally generated keys in the response message is being extended from EncryptedValue to EncryptedKey by CMP Updates [I-D.ietf-lamps-cmp-updates] also the alternative EnvelopedData can be used. In CRMF Section 2.1.9 [RFC4211] the use of EncryptedValue has been deprecated in favor of the EnvelopedData structure. Therefore, this profile specifies using EnvelopedData as specified in CMS Section 6 [RFC5652] to offer more crypto agility.

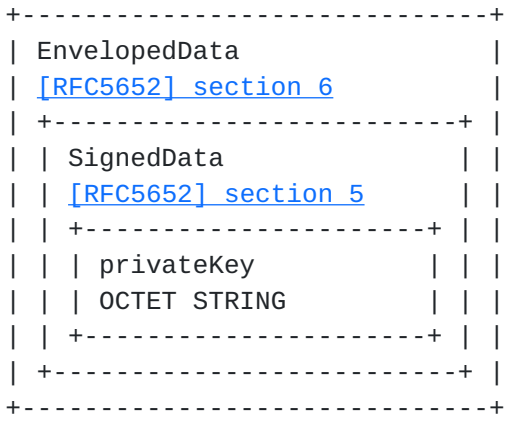

Figure 3: Encrypted private key container

The PKI management entity delivers the private key in the privateKey field in the certifiedKeyPair structure of the response message also containing the newly issued certificate.

The private key MUST be wrapped in a SignedData structure, as specified in CMS Section 5 [RFC5652], signed by the KGA generating the key pair. The signature MUST be performed using a CMP signer certificate asserting the extended key usage kp-id-cmpKGA as described in CMP Updates [I-D.ietf-lamps-cmp-updates] to show the authorization to generate key pairs on behalf of an EE.

This SignedData structure MUST be wrapped in an EnvelopedData structure, as specified in CMS Section 6 [RFC5652], encrypting it using a newly generated symmetric content-encryption key.

Note: Instead of the specification in CMP Appendix D 4.4 [RFC4210] this content-encryption key is not generated on the EE side. As we just mentioned, central key generation should only be used in this profile in case of lack of randomness on the EE.

As part of the EnvelopedData structure this content-encryption key MUST be securely provided to the EE using one of three key management techniques. The choice of the key management technique to be used by the PKI management entity depends on the authentication mechanism the EE choose to protect the request message, see CMP Updates section 3.4

[I-D.ietf-lamps-cmp-updates] for more details on which key management technique to use.

- MAC protected request message: The content-encryption key SHALL be protected using the password-based key management technique, see Section 4.1.6.3, only if the EE used MAC protection for the respected request message.

- Signature protected request message using a certificate that contains a key usage extension asserting keyAgreement: The content-encryption key SHALL be protected using the key agreement key management technique, see Section 4.1.6.1, if the certificate used by the EE for signing the respective request message contains the key usage keyAgreement. If the certificate also contains the key usage keyEncipherment, the key transport key management technique SHALL NOT be used.

- Signature protected request message using a certificate that contains a key usage extension asserting keyEncipherment: The content-encryption key SHALL be protected using the key transport key management technique, see Section 4.1.6.2, if the certificate used by the EE for signing the respective request message contains the key usage keyEncipherment and not keyAgreement.

The key agreement key management technique can be supported by most signature algorithms, as key transport key management technique can only be supported by a very limited number of algorithms. The password-based key management technique shall only be used in combination with MAC protection, which is a side-line in this document. Therefore, if central key generation is supported, the support of the key agreement key management technique is REQUIRED and the support of key transport and password-based key management techniques are OPTIONAL.

For encrypting the SignedData structure containing the private key a fresh content-encryption key MUST be generated with enough entropy with regard to the used symmetric key-encryption algorithm.

Note: Depending on the lifetime of the certificate and the criticality of the generated private key, it is advisable to use the strongest available symmetric encryption algorithm. Therefore, this specification recommends using at least AES-256.

The detailed description of the privateKey field looks like this:

```
        privateKey            OPTIONAL
    -- MUST be an EnvelopedData structure as specified in
    -- CMS [RFC5652] section 6
```

```
      version             REQUIRED
   -- MUST be set to 2
      recipientInfos      REQUIRED
   -- MUST be exactly one RecipientInfo
        recipientInfo     REQUIRED
   -- MUST be either KeyAgreeRecipientInfo (see section 5.1.5.1),
   -- KeyTransRecipientInfo (see section 5.1.5.2), or
   -- PasswordRecipientInfo (see section 5.1.5.3) is used
   -- If central key generation is supported, support of
   -- KeyAgreeRecipientInfo is REQUIRED and support of
   -- KeyTransRecipientInfo and PasswordRecipientInfo are OPTIONAL
      encryptedContentInfo
                          REQUIRED
        contentType       REQUIRED
   -- MUST be id-signedData
        contentEncryptionAlgorithm
                          REQUIRED
   -- MUST be the algorithm identifier of the symmetric
   -- content-encryption algorithm used
   -- As private keys need long-term protection, the use of AES-256
   -- or a stronger symmetric algorithm is RECOMMENDED
        encryptedContent  REQUIRED
   -- MUST be the signedData structure as specified in
   -- CMS [RFC5652] section 5 in encrypted form
          version         REQUIRED
   -- MUST be set to 3
          digestAlgorithms
                          REQUIRED
   -- MUST be exactly one digestAlgorithm identifier
            digestAlgorithmIdentifier
                          REQUIRED
   -- MUST be the OID of the digest algorithm used for generating
   -- the signature
   -- The hash algorithm used SHOULD be SHA-256
          encapContentInfo
                          REQUIRED
   -- MUST be the content that is to be signed
            contentType   REQUIRED
   -- MUST be id-data
            content       REQUIRED
   -- MUST be the privateKey as OCTET STRING
          certificates    REQUIRED
   -- SHOULD contain the certificate, for the private key used
   -- to sign the content, together with its chain
   -- If present, the first certificate in this field MUST
   -- be the certificate used for signing this content
   -- Self-signed certificates SHOULD NOT be included
   -- and MUST NOT be trusted based on the listing in any case
```

```
            crls            OPTIONAL
        -- MAY be present to provide status information on the signer or
        -- its CA certificates
            signerInfos     REQUIRED
        -- MUST be exactly one signerInfo
              version       REQUIRED
        -- MUST be set to 3
              sid           REQUIRED
                subjectKeyIdentifier
                            REQUIRED
        -- MUST be the subjectKeyIdentifier of the signer's certificate
              digestAlgorithm
                            REQUIRED
        -- MUST be the same OID as in digest algorithm
              signatureAlgorithm
                            REQUIRED
        -- MUST be the algorithm identifier of the signature algorithm
        -- used for calculation of the signature bits,
        -- like sha256WithRSAEncryption or ecdsa-with-SHA256
        -- The signature algorithm MUST be consistent with the
        -- subjectPublicKeyInfo field of the signer's certificate
              signature     REQUIRED
        -- MUST be the result of the digital signature generation
```

4.1.6.1. Using key agreement key management technique

This key management technique can be applied in combination with the PKI management operations specified in Section 4.1.1 to Section 4.1.3 using signature-based protected CMP messages. The public key of the EE certificate used for the signature-based protection of the request message MUST also be used for the Ephemeral-Static Diffie-Hellmann key establishment of the content-encryption key. To use this key management technique the KeyAgreeRecipientInfo structure MUST be used in the contentInfo field.

The KeyAgreeRecipientInfo structure included into the EnvelopedData structure is specified in CMS Section 6.2.2 [RFC5652].

The detailed description of the KeyAgreeRecipientInfo structure looks like this:

```
            recipientInfo     REQUIRED
        -- MUST be KeyAgreeRecipientInfo as specified in
             version          REQUIRED
        -- MUST be set to 3
             originator       REQUIRED
        -- MUST contain the originatorKey sequence
               algorithm      REQUIRED
        -- MUST be the algorithm identifier of the
        -- static-ephemeral Diffie-Hellmann algorithm
               publicKey      REQUIRED
        -- MUST be the ephemeral public key of the sending party
             ukm              OPTIONAL
        -- MUST be used when 1-pass ECMQV is used
             keyEncryptionAlgorithm
                              REQUIRED
        -- MUST be the same as in the contentEncryptionAlgorithm field
             recipientEncryptedKeys
                              REQUIRED
        -- MUST be exactly one recipientEncryptedKey sequence
               recipientEncryptedKey
                              REQUIRED
                 rid          REQUIRED
                   rKeyId     REQUIRED
                     subjectKeyID
                              REQUIRED
        -- MUST contain the same value as the senderKID in the
        -- respective request messages
                 encryptedKey
                              REQUIRED
        -- MUST be the encrypted content-encryption key
```

4.1.6.2. Using key transport key management technique

This key management technique can be applied in combination with the PKI management operations specified in Section 4.1.1 to Section 4.1.3 using signature-based protected CMP messages. The public key of the EE certificate used for the signature-based protection of the request message MUST also be used for key encipherment of the content-encryption key. To use this key management technique the KeyTransRecipientInfo structure MUST be used in the contentInfo field.

The KeyTransRecipientInfo structure included into the EnvelopedData structure is specified in CMS Section 6.2.1 [RFC5652].

The detailed description of the KeyTransRecipientInfo structure looks like this:

```
            recipientInfo     REQUIRED
        -- MUST be KeyTransRecipientInfo as specified in
        -- CMS section 6.2.1 [RFC5652]
                  version        REQUIRED
        -- MUST be set to 2
                  rid            REQUIRED
                    subjectKeyIdentifier
                                 REQUIRED
        -- MUST contain the same value as the senderKID in the respective
        -- request messages
                  keyEncryptionAlgorithm
                                 REQUIRED
        -- MUST contain the key encryption algorithm identifier used for
        -- public key encryption
                  encryptedKey   REQUIRED
        -- MUST be the encrypted content-encryption key
```

4.1.6.3. Using password-based key management technique

This key management technique can be applied in combination with the PKI management operation specified in Section 4.1.4 using MAC protected CMP messages. The shared secret used for the MAC protection MUST also be used for the encryption of the content-encryption key but with a different salt. To use this key management technique the PasswordRecipientInfo structure MUST be used in the contentInfo field.

The PasswordRecipientInfo structure included into the EnvelopedData structure is specified in CMS Section 6.2.3 [RFC5652].

The detailed description of the PasswordRecipientInfo structure looks like this:

```
            recipientInfo     REQUIRED
        -- MUST be PasswordRecipientInfo as specified in
        -- CMS section 6.2.4 [RFC5652]
              version         REQUIRED
        -- MUST be set to 0
              keyDerivationAlgorithm
                              REQUIRED
        -- MUST be set to id-PBKDF2 as specified in [RFC8018]
        -- The same shared secret MUST be used than used in
        -- PBMParameter data structure for the MAC protection in the
        -- header of this message
                salt          REQUIRED
        -- MUST be the random value to salt the secret key
        -- MUST be a different value than used in the PBMParameter
        -- data structure of the CMP message protection in the
        -- header of this message
                iterationCount
                              REQUIRED
        -- MUST be a limited number of times the OWF is applied
        -- To prevent brute force and dictionary attacks a reasonable
        -- high number SHOULD be used
                keyLength     REQUIRED
                prf           REQUIRED
        -- MUST be the algorithm identifier of the underlying
        -- pseudorandom function
        -- The pseudorandom function HMAC-SHA1 MUST be supported
        -- due to [RFC8018] requirements, but SHOULD NOT be used any
        -- more HMAC-SHA-256 SHOULD be used instead
              keyEncryptionAlgorithm
                              REQUIRED
        -- MUST be the same as in the contentEncryptionAlgorithm field
              encryptedKey    REQUIRED
        -- MUST be the encrypted content-encryption key
```

4.1.7. Delayed enrollment

This functional extension can be applied in combination with certificate enrollment as described in Section 4.1.1 to Section 4.1.5. The functional extension can be used in case a PKI management entity cannot respond to the certificate request in a timely manner, e.g., due to offline upstream communication or required registration officer interaction. Depending on the PKI architecture, it is not necessary that the PKI management entity directly communicating with the EE initiates the delayed enrollment.

The PKI management entity initiating the delayed enrollment MUST include the status "waiting" in the response and this response MUST NOT contain a newly issued certificate. When receiving a response with status "waiting" the EE MUST send a poll request to the PKI management entity. The PKI management entity that initiated the delayed enrollment MUST answers with a poll response containing a checkAfter time. This value indicates the minimum number of seconds that must elapse before the EE sends another poll request. As soon as the PKI management entity can provide the final response message for the initial request of the EE, it MUST provide this in response to a poll request. After receiving this response, the EE can continue the original PKI management operation as described in the respective section of this document, e.g., send a certConf message.

Typically, intermediate PKI management entities SHOULD NOT change the sender and recipient nonce even in case an intermediate PKI management entity modifies a request or a response message. In the special case of polling between EE and LRA with offline transport between an LRA and RA, see Section 5.1.4, an exception occurs. The EE and LRA exchange pollReq and pollRep messages handle the nonce words as described. When, after pollRep, the final response from the CA arrives at the LRA, the next response will contain the recipNonce set to the value of the senderNonce in the original request message (copied by the CA). The LRA needs to replace the recipNonce in this case with the senderNonce of the last pollReq because the EE will validate it in this way.

Message flow:

```
Step# EE                                    PKI management entity
 1   format ir/cr/p10cr/kur
     As described in the
       respective section
       in this document
 2                   ->ir/cr/p10cr/kur->
 3                                        handle request as described
                                            in the respective section
                                            in this document
 4                                        in case no immediate final
                                            response is possible,
                                            receive or format ip, cp
                                            or kup message containing
                                            status "waiting"
 5                    <-  ip/cp/kup  <-
 6   handle ip/cp/kup
 7   format pollReq
 8                    ->   pollReq   ->
 9                                        handle, re-protect or
                                            forward pollReq
10                                        in case the requested
                                            certificate or a
                                            corresponding response
                                            message is available,
                                            receive or format ip, cp,
                                            or kup containing the
                                            issued certificate, or
                                          format or receive pollRep
                                            with appropriate
                                            checkAfter value
11                    <-   pollRep   <-
12   handle pollRep
13   let checkAfter
       time elapse
14   continue with line 7
```

Detailed description of the first ip/cp/kup:

Response with status 'waiting' -- ip/cp/kup

Field	Value
header	
	-- MUST contain a header as described for the first response
	-- message of the respective PKI management operation

```
   body
       -- The response of the PKI management entity to the request in
       -- case no immediate appropriate response can be sent
     ip/cp/kup                   REQUIRED
       response                  REQUIRED
       -- MUST be exactly one CertResponse
         certReqId               REQUIRED
       -- MUST be set to 0
         status                  REQUIRED
       -- PKIStatusInfo structure MUST be present
           status                REQUIRED
       -- MUST be set to "waiting"
           statusString          OPTIONAL
       -- MAY be any human-readable text for debugging, logging or to
       -- display in a GUI
           failInfo              PROHIBITED
         certifiedKeyPair        PROHIBITED

   protection                    REQUIRED
       -- MUST contain protection as described for the first response
       -- message of the respective PKI management operation, but
       -- MUST use the protection key of the PKI management entity
       -- initiating the delayed enrollment and creating this response
       -- message

   extraCerts                    REQUIRED
       -- MUST contain certificates as described for the first response
       -- message of the respective PKI management operation.
       -- As no new certificate is issued yet, no respective certificate
       -- chain is included


   Polling Request -- pollReq

   Field                         Value

   header
       -- MUST contain a header as described for the certConf message
       -- of the respective PKI management operation

   body
       -- The message of the EE asks for the final response or for a
       -- time to check again
     pollReq                     REQUIRED
       certReqId                 REQUIRED
       -- MUST be exactly one value
       -- MUST be set to 0
```

```
   protection                    REQUIRED
       -- MUST contain protection as described for the certConf message
       -- of the respective PKI management operation

   extraCerts                    OPTIONAL
       -- If present, it MUST contain certificates as described for the
       -- certConf message of the respective PKI management operation


   Polling Response -- pollRep

   Field                         Value

   header
       -- MUST contain a header as described for the pkiConf message
       -- of the respective PKI management operation

   body                          pollRep
       -- The message indicated the time to after which the EE may
       -- send another pollReq messaged for this transaction
     pollRep                     REQUIRED
       -- MUST be exactly one set of the following values
       certReqId                 REQUIRED
       -- MUST be set to 0
       checkAfter                REQUIRED
       -- time in seconds to elapse before a new pollReq may be sent by
       -- the EE

   protection                    REQUIRED
       -- MUST contain protection as described for the pkiConf message
       -- of the respective profile, but
       -- MUST use the protection key of the PKI management entity that
       -- initiated the delayed enrollment and is creating this response
       -- message

   extraCerts                    OPTIONAL
       -- If present, it MUST contain certificates as described for the
       -- pkiConf message of the respective PKI management operation.


   Final response -- ip/cp/kup

   Field                         Value

   header
       -- MUST contain a header as described for the first
       -- response message of the respective PKI management operation,
       -- but the recipNonce MUST be the senderNonce of the last
```

```
    -- pollReq message

body
    -- The response of the PKI management entity to the initial
    -- request as described in the respective PKI management
    -- operation

protection                    REQUIRED
    -- MUST contain protection as described for the first response
    -- message of the respective PKI management operation, but
    -- MUST use the protection key of the PKI management entity that
    -- initiated the delayed enrollment and forwarding the response
    -- message

extraCerts                    REQUIRED
    -- MUST contain certificates as described for the first
    -- response message of the respective PKI management operation
```

4.2. Revoking a certificate

This PKI management operation should be used by an entity to request the revocation of a certificate. Here the revocation request is used by an EE to revoke one of its own certificates. A PKI management entity could also act as an EE to revoke one of its own certificates.

The revocation request message MUST be signed using the certificate that is to be revoked to prove the authorization to revoke to the PKI. The revocation request message is signature-protected using this certificate.

An EE requests the revocation of an own certificate at the CA that issued this certificate. The PKI management entity responds with a message that contains the status of the revocation from the CA.

Preconditions:

1 The certificate the EE wishes to revoke is not yet expired or revoked.

Message flow:

```
Step# EE                                   PKI management entity
  1   format rr
  2                      ->   rr      ->
  3                                        handle, re-protect or
                                             forward rr
  4                                        receive rp
  5                      <-   rp      <-
  6   handle rp
```

For this PKI management operation, the EE MUST include exactly one RevDetails structure in the rr message body. In case no error occurred the response to the rr MUST be a rp message. The PKI management entity MUST produce a rp containing a status field with a single set of values.

Detailed message description:

Revocation Request -- rr

```
Field                          Value

header
    -- As described in section 3.1

body
    -- The request of the EE to revoke its certificate
  rr                           REQUIRED
    -- MUST contain exactly one element of type RevDetails
    -- If more revocations are desired, further requests MUST be
    -- packaged in separate PKI Messages
    certDetails                REQUIRED
    -- MUST be present and is of type CertTemplate
      serialNumber             REQUIRED
    -- MUST contain the certificate serialNumber attribute of the
    -- X.509 certificate to be revoked
      issuer                   REQUIRED
    -- MUST contain the issuer attribute of the X.509 certificate to
    -- be revoked
    crlEntryDetails            REQUIRED
    -- MUST contain exactly one reasonCode of type CRLReason (see
    -- [RFC5280] section 5.3.1)
    -- If the reason for this revocation is not known or shall not be
    -- published the reasonCode MUST be 0 = unspecified

protection                     REQUIRED
```

```
    -- As described in section 3.2 and the private key related to the
    -- certificate to be revoked

extraCerts                    REQUIRED
    -- As described in section 3.3


Revocation Response -- rp

Field                         Value

header
    -- As described in section 3.1

body
    -- The responds of the PKI management entity to the request as
    -- appropriate
  rp                          REQUIRED
    status                    REQUIRED
    -- MUST contain exactly one element of type PKIStatusInfo
      status                  REQUIRED
    -- positive value allowed: "accepted"
    -- negative value allowed: "rejection"
      statusString            OPTIONAL
    -- MAY be any human-readable text for debugging, logging or to
    -- display in a GUI
      failInfo                OPTIONAL
    -- MAY be present if and only if status is "rejection"

protection                    REQUIRED
    -- As described in section 3.2

extraCerts                    REQUIRED
    -- As described in section 3.3
```

4.3. Error reporting

This functionality should be used by an EE to report any error conditions upstream to the PKI management entity. Error reporting by a PKI management entity downstream to the EE is described in Section 5.3.

In case the error condition is related to specific details of an ip, cp, or kup response message and a confirmation is expected the error condition MUST be reported in the respective certConf message with negative contents.

General error conditions, e.g., problems with the message header, protection, or extraCerts, and negative feedback on rp, pollRep, or pkiConf messages MAY be reported in the form of an error message.

In both situations the EE reports error in the PKIStatusInfo structure of the respective message.

Depending on the PKI architecture, the PKI management entity MUST forward the error message (upstream) to the next PKI management entity and MUST terminate this PKI management operation.

The PKIStatusInfo structure is used to report errors. The PKIStatusInfo structure SHOULD consist of the following fields:

- status: Here the PKIStatus value "rejection" is the only one allowed.

- statusString: Here any human-readable valid value for logging or to display in a GUI SHOULD be added.

- failInfo: Here the PKIFailureInfo values MAY be used in the following way. For explanation of the reason behind a specific value, please refer to [RFC4210] Appendix F.

  * transactionIdInUse: This is sent by a PKI management entity in case the received request contains a transaction ID that is already in use for another transaction. An EE receiving such error message SHOULD resend the request in a new transaction using a different transaction ID.

  * systemUnavail or systemFailure: This is sent by a PKI management entity in case a back-end system is not available or currently not functioning correctly. An EE receiving such error message SHOULD resend the request in a new transaction after some time.

Detailed error message description:

Error Message -- error

Field	Value
header	
-- As described in section 3.1	
body	
-- The message sent by the EE or the (L)RA/CA to indicate an	
-- error that occurred	
error	REQUIRED
pKIStatusInfo	REQUIRED
status	REQUIRED
-- MUST have the value "rejection"	
statusString	RECOMMENDED
-- SHOULD be any human-readable text for debugging, logging	
-- or to display in a GUI	
failInfo	OPTIONAL
-- MAY be present	
protection	REQUIRED
-- As described in section 3.2	
extraCerts	OPTIONAL
-- As described in section 3.3	

4.4. Support messages

The following support messages offer on demand in-band transport of content that may be provided by the PKI management entity and relevant to the EE. The general messages and general response are used for this purpose. Depending on the environment, these requests may be answered by the LRA, RA, or CA.

The general message and general response transport InfoTypeAndValue structures. In addition to those infoType values defined in CMP [RFC4210] further OIDs MAY be defined to define new PKI management operations, or general-purpose support messages as needed in a specific environment.

Content specified in this document is describs the following:

- Request of CA certificates

- Update of Root CA certificates

o Parameters needed for a planned certificate request message

4.4.1. General message and response

The PKI management operation is similar to that given in CMP Appendix E.5 [RFC4210]. In this section the general message (genm) and general response (genp) are described. The specific InfoTypeAndValue structures are described in the following sections.

The behavior in case an error occurs is described in Section 4.3.

Message flow:

```
Step# EE                                    PKI management entity
 1   format genm
 2                      ->   genm    ->
 3                                        handle, re-protect or
                                            forward genm
 4                                        format or receive genp
 5                      <-   genp    <-
 6   handle genp
```

Detailed message description:

General Message -- genm

```
Field                         Value

header
    -- As described in section 3.1

body
    -- The request of the EE to receive information
  genm                        REQUIRED
    -- MUST contain exactly one element of type
    -- InfoTypeAndValue
    infoType                  REQUIRED
    -- MUST be the OID identifying the specific PKI
    -- management operation described below
    infoValue                 OPTIONAL
    -- MUST be as described in the specific PKI
    -- management operation described below

protection                    REQUIRED
    -- As described in section 3.2

extraCerts                    REQUIRED
    -- As described in section 3.3
```

```
   General Response -- genp

   Field                          Value

   header
       -- As described in section 3.1

   body
       -- The response of the PKI management entity to the
       -- information request
     genp                         REQUIRED
       -- MUST contain exactly one element of type
       -- InfoTypeAndValue
       infoType                   REQUIRED
       -- MUST be the OID identifying the specific PKI
       -- management operation described below
       infoValue                  OPTIONAL
       -- MUST be as described in the specific PKI
       -- management operation described below

   protection                     REQUIRED
       -- As described in section 3.2

   extraCerts                     REQUIRED
       -- As described in section 3.3
```

4.4.2. Get CA certificates

This PKI management operation can be used by an EE to request CA certificates from the PKI management entity.

An EE requests CA certificates from the PKI management entity by sending a general message with OID id-it-caCerts. The PKI management entity responds with a general response with the same OID that either contains a SEQUENCE of certificates populated with the available CA intermediate and issuing CA certificates or with no content in case no CA certificate is available.

The message sequence for this PKI management operation is as given in Section 4.4.1, with the following specific content:

1 the body MUST contain as infoType the OID id-it-caCerts

2 the infoValue of the request MUST be absent

3 if present, the infoValue of the response MUST be caCerts field

The infoValue field of the general response containing the id-it-caCerts OID looks like this:

```
      infoValue               OPTIONAL
    -- MUST be absent if no CA certificate is available
    -- MUST be present if CA certificates are available
    -- MUST be a sequence of CMPCertificate
```

4.4.3. Get root CA certificate update

This PKI management operation can be used by an EE to request an update of an existing root CA Certificate by the EE.

An EE requests a root CA certificate update from the PKI management entity by sending a general message with OID id-it-rootCaKeyUpdate as infoType and no infoValue. The PKI management entity responds with a general response with the same OID that either contains the update of the root CA certificate consisting of up to three certificates, or with no content in case no update is available.

These three certificates are described in more detail in section 4.4.1, section 6.2, and Appendix E.3 of [RFC4210]. The newWithNew certificate is the new root CA certificates and is REQUIRED to be present in the response message. The newWithOld certificate is RECOMMENDED to be present in the response message though it is REQUIRED for those cases where the receiving entity trusts the old root CA certificate and wishes to gain trust in the new root CA certificate. The oldWithNew certificate is OPTIONAL though it is only needed in a scenario where the requesting entity already trusts the new root CA certificate and wants to gain trust in the old root certificate.

The message sequence for this PKI management operation is as given in Section 4.4.1, with the following specific content:

1 the body MUST contain as infoType the OID id-it-rootCaKeyUpdate

2 the infoValue of the request MUST be absent

3 if present, the infoValue of the response MUST be a RootCaKeyUpdate structure

The infoValue field of the general response containing the id-it-rootCaKeyUpdate extension looks like this:

```
      infoValue               OPTIONAL
    -- MUST be absent if no update of the root CA certificate is
    -- available
    -- MUST be present if an update of the root CA certificate
    -- is available and MUST be of type RootCaKeyUpdate
        newWithNew            REQUIRED
    -- MUST be present if infoValue is present
    -- MUST contain the new root CA certificate
        newWithOld            RECOMMENDED
    -- SHOULD be present if infoValue is present
    -- MUST contain an X.509 certificate containing the new public
    -- root CA key signed with the old private root CA key
        oldWithNew            OPTIONAL
    -- MAY be present if infoValue is present
    -- MUST contain an X.509 certificate containing the old public
    -- root CA key signed with the new private root CA key
```

4.4.4. Get certificate request template

This PKI management operation can be used by an EE to request a template with parameters for a future certificate request operation.

An EE requests certificate request parameters from the PKI management entity by sending a general message with OID id-it-certReqTemplate. The PKI management entity responds with a general response with the same OID that either contains a certificate template with the required fields and optionally a rsaKeyLen field containing requirements on, e.g., algorithm identifier for key pair generation or certificate fields and extensions, or with no content in case no specific requirements are made by the PKI.

The EE SHOULD follow the requirements from the received CertTemplate and the optional rsaKeyLen fields, by filling in all the fields requested and taking over all the field values provided. The EE SHOULD NOT add further CertTemplate fields, Name components, and extensions or their (sub-)components.

Note: We deliberately do not use 'MUST' or 'MUST NOT' here in order to allow more flexibility in case the rules given here are not sufficient for specific scenarios. The EE can populate the certificate request as wanted and ignore any of the requirements contained in the CertReqTemplate response message. On the other hand, a PKI management entity is free to ignore or replace the content of the certificate request provided by the EE. The

CertReqTemplate PKI management operation offers means to ease a joint understanding which fields should be used.

In case a field of type Name, e.g., issuer or subject name, is present but has the value NULL-DN (i.e., has an empty list of RDN components) the field SHOULD be included with content provided by the EE. Similarly, in case an X.509v3 extension is present but its extnValue is empty this means that the extension SHOULD be included with content provided by the EE. In case a Name component, for instance a common name or serial number, is given but has an empty string value the EE SHOULD fill in a value. Similarly, in case an extension has sub-components (e.g., an IP address in a SubjectAltName field) with empty value, the EE SHOULD fill in a value.

The EE MUST ignore (i.e., not include and fill in) empty fields, extensions, and sub-components that it does not know.

If the publicKey field of type SubjectPublicKeyInfo is present its algorithm field specifies the type of the public key to request a certificate for. The algorithm field contains the key type OID of the public key. For EC keys the full curve information MUST be specified as described in the respective standard documents. For RSA keys the key length MUST be specified in the rsaKeyLen field of the outer infoValue field. The algorithm field MUST be followed by a zero-length BIT STRING for the subjectPublicKey. If the publicKey field is not present the EE is free to choose the public key type and parameters.

In the certTemplate structure the serialNumber, signingAlg, issuerUID, and subjectUID fields MUST be omitted.

The message sequence for this PKI management operation is as given in Section 4.4.1, with the following specific content:

1 the body MUST contain as infoType the OID id-it-certReqTemplate

2 the infoValue of the request MUST be absent

3 if present, the infoValue of the response MUST be a SEQUENCE of a certTemplate structure and an rsaKeyLen field of type INTEGER

The infoValue field of the general response containing the id-it-certReqTemplate OID looks like this:

```
      InfoValue               OPTIONAL
    -- MUST be absent if no requirements are available
    -- MUST be present if the PKI management entity has any
    -- requirements on the content of the certificates template
    -- is available and MUST be of type CertReqTemplateValue
        certTemplate          REQUIRED
    -- MUST be present if infoValue is present
    -- MUST contain the prefilled certTemplate structure elements
        rsaKeyLen             OPTIONAL
    -- This field is of type INTEGER. Any reasonable RSA key length
    -- MUST be specified if the algorithm in the
    -- subjectPublicKeyInfo field of the certTemplate has the OID
    -- rsaEncryption.
    -- MUST be omitted in otherwise.
```

5. LRA and RA focused PKI management operations

This chapter focuses on the communication among different PKI management entities. Depending on the network and PKI solution design, these will either be an LRA, RA or CA.

Typically, a PKI management entity forwards messages from downstream, but it may also reply to them itself. Besides forwarding of received messages a PKI management entity could also need to revoke certificates of EEs, report errors, or may need to manage its own certificates.

5.1. Forwarding of messages

Each CMP request message (i.e., ir, cr, p10cr, kur, pollReq, or certConf) or error message coming from an EE or the previous (downstream) PKI management entity MUST be sent to the next (upstream) PKI management entity. This PKI management entity MUST forward response messages to the next (downstream) PKI management entity or EE.

The PKI management entity SHOULD verify the protection, the syntax, the required message fields, the message type, and if applicable the authorization and the proof-of-possession of the message. Additional checks or actions MAY be applied depending on the PKI solution requirements and concept. If one of these verification procedures fails, the (L)RA SHOULD respond with a negative response message and SHOULD not forward the message further upstream. General error conditions should be handled as described in Section 4.3 and Section 5.3.

A PKI management entity SHOULD not change the received message if not necessary. The PKI management entity SHOULD only update the message protection if it is technically necessary. Concrete PKI system specifications may define in more detail if and when to do so.

This is particularly relevant in the upstream communication of a request message.

Each hop in a chain of PKI management entity has one or more functionalities, e.g., a PKI management entity

- may need to verify the identities of EEs or base authorization decisions for certification request processing on specific knowledge of the local setup, e.g., by consulting an inventory or asset management system,
- may need to add fields to certificate request messages,
- may need to store data from a message in a database for later usage or documentation purposes,
- may provide traversal of a network boundary,
- may need to double-check if the messages transferred back and forth are properly protected and well formed,
- may provide a proof that it has performed all required checks,
- may initiate a delayed enrollment due to offline upstream communication or registration officer interaction,
- may grant the request of an EE to omit sending a confirmation message, or
- can collect messages from different LRAs and forward them to the CA.

Therefore, the decision if a message should be forwarded

- unchanged with the original protection,
- unchanged with a new protection, or
- changed with a new protection

depends on the PKI solution design and the associated security policy (CP/CPS [RFC3647]).

This section specifies the different options a PKI management entity may implement and use.

A PKI management entity MAY update the protection of a message

- if it performs changes to the header or the body of the message,
- if it needs to prove checks or validations performed on the message to one of the next (upstream) PKI components,
- if it needs to protect the message using a key and certificate from a different PKI, or
- if it needs to replace a MAC based-protection.

This is particularly relevant in the upstream communication of certificate request messages.

The message protection covers only the header and the body and not the extraCerts. The PKI management entity MAY change the extraCerts in any of the following message adaptations, e.g., to sort or add needed or to delete needless certificates to support the next hop. This may be particularly helpful to extend upstream messages with additional certificates or to reduce the number of certificates in downstream messages when forwarding to constrained devices.

5.1.1. Not changing protection

This alternative to forward a message can be used by any PKI management entity to forward an original CMP message without changing the header, body or protection. In any of these cases the PKI management entity acts more like a proxy, e.g., on a network boundary, implementing no specific RA-like security functionality to the PKI.

This alternative to forward a message MUST be used for forwarding kur messages that must not be approved by the respective PKI management entity.

5.1.2. Replacing protection

The following two alternatives to forward a message can be used by any PKI management entity to forward a CMP message with or without changes, but providing its own protection using its CMP signer key to assert approval of this message. In this case the PKI management entity acts as an actual Registration Authority (RA), which implements important security functionality of the PKI.

Before replacing the existing protection by a new protection, the PKI management entity MUST verify the protection provided by the EE or by the previous PKI component and approve its content including any own modifications. For certificate requests the PKI management entity MUST verify in particular the included proof-of-possession self-signature of the certTemplate using the public key of the requested certificate and MUST check that the EE, as authenticated by the message protection, is authorized to request a certificate with the subject as specified in the certTemplate.

In case the received message has been protected by a CA or another PKI management entity, the current PKI management entity MUST verify its protection and approve its content including any own modifications. For certificate requests the PKI management entity MUST check that the other PKI management entity, as authenticated by the protection of the incoming message, was authorized to issue or forward the request.

These message adaptations MUST NOT be applied to kur request messages as described in Section 4.1.3 since their original protection using the key and certificate to be updated needs to be preserved, unless the regCtrl OldCertId is used to clearly identify the certificate to be updated.

5.1.2.1. Keeping proof-of-possession

This alternative to forward a message can be used by any PKI management entity to forward a CMP message with or without modifying the message header or body while preserving any included proof-of-possession.

By replacing the existing protection using its own CMP signer key the PKI management entity provides a proof of verifying and approving of the message as described above.

In case the PKI management entity modifies the certTemplate of an ir or cr message, the message adaptation in Section 5.1.2.2 needs to be applied instead.

5.1.2.2. Breaking proof-of-possession

This alternative to forward a message can be used by any PKI management entity to forward an ir or cr message with modifications of the certTemplate i.e., modification, addition, or removal of fields. Such changes will break the proof-of-possession provided by the EE in the original message.

By replacing the existing using its own CMP signer key the PKI management entity provides a proof of verifying and approving the new message as described above.

In addition to the above the PKI management entity MUST verify in particular the proof-of-possession contained in the original message as described above. If these checks were successfully performed the PKI management entity MUST change the popo to raVerified.

The popo field MUST contain the raVerified choice in the certReq structure of the modified message as follows:

```
popo
  raVerified              REQUIRED
-- MUST have the value NULL and indicates that the PKI
-- management entity verified the popo of the original
-- message
```

5.1.3. Adding Protection

This PKI management operation can be used by a PKI management entity to add another protection to one or several PKI management messages.

The nested message is a PKI management message containing a PKIMessages sequence as its body containing one or more CMP messages.

As specified in the updated Section 5.1.3.4 of RFC4210 [RFC4210] (see Section 3.3 of CMP Updates [I-D.ietf-lamps-cmp-updates]) there are different use case for adding another protection by a PKI management entity. Specific procedures are described in more detail in the following sections.

The behavior in case an error occurs is described in Section 4.3.

Message flow:

```
Step# PKI management entity              PKI management entity
 1   format nested
 2                      ->  nested   ->
 3                                       handle, re-protect or
                                           forward nested
 4                                       format or receive nested
 5                      <-  nested   <-
 6   handle nested
```

Detailed message description:

Nested Message - nested

Field	Value
header	
-- As described in section 3.1	
body	nested
-- Container to provide additional protection to original	
-- messages and to bundle request or response messages	
PKIMessages	REQUIRED
-- MUST be a sequence of one or more CMP messages	
protection	REQUIRED
-- As described in section 3.2 using the CMP signer key of	
-- the PKI management entity	
extraCerts	REQUIRED
-- As described in section 3.3	

5.1.3.1. Handling a single PKI management message

A PKI management entity may prove successful validation and authorization of a PKI management message by adding an additional signature to the original PKI management message.

A PKI management entity SHALL wrap the original PKI management messages in a nested message structure. The additional signature as prove of verification and authorization by the PKI management entity MUST be applies as signature-based message protection of the nested message.

5.1.3.2. Handling a batch of PKI management messages

A PKI management entity MAY bundle any number of PKI management messages for batch processing or to transfer a bulk of PKI management messages via an offline interface using the nested message structure. The nested message can be either used on the upstream interface towards the next PKI management entity as well as on the downstream interface from the PKI management entity towards the EE.

This PKI management operation is typically used on the interface between LRA and RA to bundle several PKI management messages for offline transport. In this case the EE needs to make use of delayed enrollment as described in Section 4.1.7. If the RA may need different routing information per nested PKI management message a

suitable mechanism may need to be implemented. This mechanism strongly depends on the requirements of the target architecture; therefore, it is out of scope of this document.

An initial nested message is generated locally at the PKI management entity. For the initial nested message, the PKI management entity acts as a protocol end point and therefore a fresh transactionId and a fresh senderNonce MUST be used in the header of the nested message. The recipient field MUST identify the PKI management entity that is expected to unpack the nested message. An initial nested message should contain only request messages, e.g., ir, cr, p10cr, kur, certConf, rr, or genm. While building the initial nested message the PKI management entity SHOULD store the transactionIds and the senderNonces of all bundled messages together with the transactionId of the initial nested message.

Such an initial nested message is sent to the next PKI management entity and SHOULD be answered with a responding nested message. This responding message SHOULD use the transactionId of the initial nested message and return the senderNonce of the initial nested message as recipNonce of the responding nested message. The responding nested message SHOULD bundle one response message (e.g. ip, cp, kup, pkiconf, rp, genp, error) for each request message (i.e., for each transactionId) in the initial nested message. While unbundling the responding nested message it is possible to determine lost and unexpected responses based on the previously stored transactionIds and senderNonces. While forwarding the unbundled responses, odd messages SHOULD be dropped, and lost messages should be replaced by an error message to inform the EE about the failed certificate management operation.

The PKI management entity building the nested message applies a signature-based protection using its CMP-signer key as transport protection. This protection SHALL NOT be regarded as prove of verification or authorization of the bundled PKI management messages.

5.1.4. Initiating delayed enrollment

This functional extension can be used by a PKI management entity to initiate delayed enrollment. In this case a PKI management entity MUST add the status waiting in the response message. The PKI management entity MUST then reply to the pollReq messages as described in Section 4.1.7.

5.2. Revoking certificates on behalf of another's entities

This PKI management operation can be used by a PKI management entity to revoke a certificate of any other entity. This revocation request message MUST be signed by the PKI management entity using its own CMP signer key to prove to the PKI authorization to revoke the certificate on behalf of the EE.

Preconditions:

1 the certificate to be revoked MUST be known to the PKI management entity

2 the PKI management entity MUST have the authorization to revoke the certificates of other entities issued by the corresponding CA

The message sequence for this PKI management operation is identical to that given in Section 4.2, with the following changes:

1 it is not required that the certificate to be revoked is not yet expired or revoked

2 the PKI management entity acts as EE for this message exchange

3 the rr messages MUST be signed using the CMP signer key of the PKI management entity.

5.3. Error reporting

This functionality should be used by the PKI management entity to report any error conditions downstream to the EE. Potential error reporting by the EE upstream to the PKI management entity is described in Section 4.3.

In case the error condition is related to specific details of an ir, cr, p10cr, or kur request message it MUST be reported in the specific response message, i.e., an ip, cp, or kup with negative contents.

General error conditions, e.g., problems with the message header, protection, or extraCerts, and negative feedback on rr, pollReq, certConf, or error messages MUST be reported in the form of an error message.

In both situations the PKI management entity reports the errors in the PKIStatusInfo structure of the respective message as described in Section 4.3.

An EE receiving any such negative feedback SHOULD log the error appropriately and MUST terminate the current transaction.

6. CMP message transport variants

The CMP messages are designed to be self-contained, such that in principle any transport can be used. HTTP SHOULD be used for online transport while file-based transport MAY be used in case offline transport is required. In case HTTP transport is not desired or possible, CMP messages MAY also be piggybacked on any other reliable transport protocol, e.g., CoAP [RFC7252].

Independently of the means of transport it could happen that messages are lost, or a communication partner does not respond. In order to prevent waiting indefinitely, each CMP client component SHOULD use a configurable per-request timeout, and each CMP server component SHOULD use a configurable per-response timeout in case a further message is to be expected from the client side. In this way a hanging transaction can be closed cleanly with an error and related resources (for instance, any cached extraCerts) can be freed.

When conveying a CMP messages in HTTP or MIME-based transport protocols the internet media type "application/pkixcmp" MUST be set for transport encoding as specified in RFC2510 in Section 5.3 [RFC2510] and RFC6712 in Section 3.4 [RFC7712].

6.1. Definition and discovery of HTTP URIs

Each PKI management entity supporting HTTP or HTTPS transport MUST support the use of the path-prefix of '/.well-known/' as defined in [RFC5785] and the registered name of 'cmp' to ease interworking in a multi-vendor environment.

The CMP client MUST be configured with sufficient information to form the CMP server URI. This MUST be at least the authority portion of the URI, e.g., 'www.example.com:80', or the full operational path of the PKI management entity. An additional arbitrary label, e.g., 'arbitraryLabel', MAY be configured as a separate component or as part of the full operational path to provide further information to address multiple CAs or certificate profiles. A valid full operational path can look like this:

1 http://www.example.com/.well-known/cmp

2 http://www.example.com/.well-known/cmp/keyupdate

3 http://www.example.com/.well-known/cmp/arbitraryLabel

4 http://www.example.com/.well-known/cmp/arbitraryLabel/keyupdate

PKI management operations SHOULD use the following URI path:

PKI management operation	Path	Details
Enroll client to new PKI (REQUIRED)	/initialization	Section 4.1.1
Enroll client to existing PKI (OPTIONAL)	/certification	Section 4.1.2
Update client certificate (REQUIRED)	/keyupdate	Section 4.1.3
Enroll client using PKCS#10 (OPTIONAL)	/p10	Section 4.1.5
Enroll client using central key generation (OPTIONAL)	/serverkeygen	Section 4.1.6
Revoke client certificate (RECOMMENDED)	/revocation	Section 4.2
Get CA certificates (OPTIONAL)	/getcacert	Section 4.4.2
Get root CA certificate update (OPTIONAL)	/getrootupdate	Section 4.4.3
Get certificate request template (OPTIONAL)	/getcertreqtemplate	Section 4.4.4
Additional protection (OPTIONAL)	/nested	Section 5.1.3

Table 1: HTTP endpoints

Subsequent certConf, error, and pollReq messages are sent to the URI of the respective PKI management operation.

The discovery of supported endpoints as defined above will provide the information to the EE, how to contact the PKI management entity and, if available, how to request enrolment for a specific certificate profile or revoke a certificate at a specific CA.

Querying the PKI management entity, the EE will get a list of potential endpoints supported by the PKI management entity.

Performing a GET on "/.well-known/cmp" to the default port returns a set of links to endpoints available from the server or RA. In addition to the link also the expected format of the data object is provided as content type (ct).

The following provides an illustrative example for a PKI management entity supporting different PKI management operations for a single certificate profile or a single CA.

Detailed message description:

REQ: GET /.well-known/cmp

RES: Content
</cmp/initialization>;ct=pkixcmp
</cmp/certification >;ct=pkixcmp
</cmp/keyupdate >;ct=pkixcmp
</cmp/p10>;ct=pkixcmp
</cmp/revocation>;ct=pkixcmp
</cmp/ca2/revocation>;ct=pkixcmp
</cmp/getcacerts>;ct=pkixcmp
</cmp/getrootupdate>;ct=pkixcmp
</cmp/getcertreqtemplate >;ct=pkixcmp

As it is very likely, that a CA supports different certification profiles or that the RA offers PKI management operations for different issuing CAs, the discovery can also be used to provide the information about these options. The second example listing contains the supported PKI management operations for three different certificate profiles. The supported CA hierarchy consists of one root CA and two issuing CAs.

Detailed message description:

REQ: GET /.well-known/cmp

RES: Content
</cmp/certprofile1/initialization>;ct=pkixcmp
</cmp/certprofile2/initialization>;ct=pkixcmp
</cmp/certprofile3/initialization>;ct=pkixcmp
</cmp/certprofile1/certification >;ct=pkixcmp
</cmp/certprofile2/certification >;ct=pkixcmp
</cmp/certprofile3/certification >;ct=pkixcmp
</cmp/certprofile1/keyupdate >;ct=pkixcmp
</cmp/certprofile2/keyupdate >;ct=pkixcmp
</cmp/certprofile3/keyupdate >;ct=pkixcmp
</cmp/certprofile1/p10>;ct=pkixcmp
</cmp/certprofile2/p10>;ct=pkixcmp
</cmp/certprofile3/p10>;ct=pkixcmp
</cmp/ca1/revocation>;ct=pkixcmp
</cmp/ca2/revocation>;ct=pkixcmp
</cmp/getcacerts>;ct=pkixcmp
</cmp/rootca1/getrootupdate>;ct=pkixcmp
</cmp/certprofile1/getcertreqtemplate >;ct=pkixcmp
</cmp/certprofile2/getcertreqtemplate >;ct=pkixcmp
</cmp/certprofile3/getcertreqtemplate >;ct=pkixcmp

There are different options in the handling of the naming. The PKI management entity either needs to offer the certprofile or CA labels the EE expects. Alternatively, a mechanism is required to configure this information to the EE beforehand.

6.2. HTTP transport

This transport mechanism can be used by a PKI entity to transfer CMP messages over HTTP. If HTTP transport is used the specifications as described in [RFC6712] MUST be followed.

6.3. HTTPS transport using certificates

This transport mechanism can be used by a PKI entity to further protect the HTTP transport as described in Section 6.2 using TLS 1.2 [RFC5246] or TLS 1.3 [RFC8446] as described in [RFC2818] with certificate-based authentication. Using this transport mechanism, the CMP transport via HTTPS MUST use TLS server authentication and SHOULD use TLS client authentication.

EE:

- The EE SHOULD use a TLS client certificate as far as available. If no dedicated TLS certificate is available, the EE SHOULD use an already existing certificate identifying the EE (e.g., a manufacturer certificate).

- If no TLS certificate is available at the EE, server-only authenticated TLS SHOULD be used.

- The EE MUST validate the TLS server certificate of its communication partner.

PKI management entity:

- Each PKI management entity SHOULD use a TLS client certificate on its upstream (client) interface.

- Each PKI management entity MUST use a TLS server certificate on its downstream (server) interface.

- Each PKI management entity MUST validate the TLS certificate of its communication partners.

NOTE: The requirements for checking certificates given in [RFC5280], [RFC5246] and [RFC8446] MUST be followed for the TLS layer. Certificate status checking SHOULD be used for the TLS certificates of communication partners.

6.4. HTTPS transport using shared secrets

This transport mechanism can be used by a PKI entity to further protect the HTTP transport as described in Section 6.2 using TLS 1.2 [RFC5246] or TLS 1.3 [RFC8446] as described in [RFC2818] with mutual authentication based on shared secrets as described in [RFC5054].

EE:

- The EE MUST use the shared symmetric key for authentication.

PKI management entity:

- The PKI management entity MUST use the shared symmetric key for authentication.

6.5. Offline transport

For transporting CMP messages between PKI entities any mechanism can be used that is able to store and forward binary objects of

sufficient length and with sufficient reliability while preserving the order of messages.

The transport mechanism SHOULD be able to indicate message loss, excessive delay, and possibly other transmission errors. In such cases the PKI entities using this mechanism SHOULD report an error as specified in Section 4.3.

6.5.1. File-based transport

CMP messages MAY be transferred between PKI entities using file-system-based mechanisms, for instance when an off-line end entity or a PKI management entity performs delayed enrollment. Each file MUST contain the ASN.1 DER encoding of one CMP message only. There MUST be no extraneous header or trailer information in the file. The file type extensions ".PKI" SHOULD be used.

6.5.2. Other asynchronous transport protocols

Other asynchronous transport protocols, e.g., email or website up-/download, MAY transfer CMP messages between PKI entities. A MIME wrapping is defined for those environments that are MIME native. The MIME wrapping in this section is specified in [RFC8551], section 3.1.

The ASN.1 DER encoding of the CMP messages MUST be transferred using the "application/pkixcmp" content type and base64-encoded content-transfer-encoding as specified in [RFC2510], section 5.3. A filename MUST be included either in a content-type or a content-disposition statement. The extension for the file MUST be ".PKI".

6.6. CoAP transport

In constrained environments where no HTTP transport is desired or possible, CoAP [RFC7252] as specified in [I-D.msahni-tbd-cmpv2-coap-transport] MAY be used instead.

6.7. Piggybacking on other reliable transport

For online transfer where no HTTP transport is desired or possible CMP messages MAY also be transported on some other reliable protocol. Connection and error handling mechanisms like those specified for HTTP in [RFC6712] need to be implemented.

Such specification is out of scope of this document and would need to be specifies in a separate document, e.g., in the scope of the respective transport protocol used.

7. IANA Considerations

< TBD: The OID id-it-caCerts, id-it-rootCaKeyUpdate, and id-it-certReqTemplate are not yet defined and should be registered in the tree 1.3.6.1.5.5.7.4 (id-it) like other infoType OIDs, see CMP Appendix F [RFC4210] on page 92. >

8. Security Considerations

< TBD: Add any security considerations >

9. Acknowledgements

We would like to thank the various reviewers of this document.

10. References

10.1. Normative References

[I-D.ietf-lamps-cmp-updates] Brockhaus, H., "CMP Updates", draft-ietf-lamps-cmp-updates-02 (work in progress), July 2020.

[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>.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, DOI 10.17487/RFC4210, September 2005, <https://www.rfc-editor.org/info/rfc4210>.

[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>.

[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>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <https://www.rfc-editor.org/info/rfc5785>.

[RFC6712] Kause, T. and M. Peylo, "Internet X.509 Public Key Infrastructure -- HTTP Transfer for the Certificate Management Protocol (CMP)", RFC 6712, DOI 10.17487/RFC6712, September 2012, <https://www.rfc-editor.org/info/rfc6712>.

10.2. Informative References

[ETSI-3GPP]
3GPP, "TS33.310; Network Domain Security (NDS); Authentication Framework (AF); Release 16; V16.1.0", December 2018, <http://www.3gpp.org/ftp/Specs/archive/33_series/33.310/>.

[I-D.msahni-tbd-cmpv2-coap-transport]
Sahni, M., "CoAP Transport for CMPV2", draft-msahni-tbd-cmpv2-coap-transport-00 (work in progress), June 2020.

[IEC62443-3-3]
IEC, "Industrial communication networks - Network and system security - Part 3-3: System security requirements and security levels", IEC 62443-3-3, August 2013, <https://webstore.iec.ch/publication/7033>.

[IEEE802.1AR]
IEEE, "802.1AR Secure Device Identifier", June 2018, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[NIST-CSFW] NIST, "Framework for Improving Critical Infrastructure Cybersecurity Version 1.1", April 2018, <https://www.nist.gov/publications/framework-improving-critical-infrastructure-cybersecurity-version-11>.

[RFC2510] Adams, C. and S. Farrell, "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, DOI 10.17487/RFC2510, March 1999, <https://www.rfc-editor.org/info/rfc2510>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

[RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 3647, DOI 10.17487/RFC3647, November 2003, <https://www.rfc-editor.org/info/rfc3647>.

[RFC5054] Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, "Using the Secure Remote Password (SRP) Protocol for TLS Authentication", RFC 5054, DOI 10.17487/RFC5054, November 2007, <https://www.rfc-editor.org/info/rfc5054>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7712] Saint-Andre, P., Miller, M., and P. Hancke, "Domain Name Associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP)", RFC 7712, DOI 10.17487/RFC7712, November 2015, <https://www.rfc-editor.org/info/rfc7712>.

[RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <https://www.rfc-editor.org/info/rfc8366>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8551] Schaad, J., Ramsdell, B., and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification", RFC 8551, DOI 10.17487/RFC8551, April 2019, <https://www.rfc-editor.org/info/rfc8551>.

[UNISIG] UNISIG, "Subset-137; ERTMS/ETCS On-line Key Management FFFIS; V1.0.0", December 2015, <https://www.era.europa.eu/filebrowser/download/542_en>.

Appendix A. ASN.1 Syntax

```
id-it-caCerts OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 4 xxx}
CaCerts ::= SEQUENCE OF CMPCertificate
}

id-it-rootCaKeyUpdate OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 4 xxx}
RootCaKeyUpdate ::= SEQUENCE {
    newWithNew       CMPCertificate
    newWithOld   [0] CMPCertificate OPTIONAL,
    oldWithNew   [1] CMPCertificate OPTIONAL,
}

id-it-certReqTemplate OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 4 xxx}
CertReqTemplateValue ::= SEQUENCE {
    certTemplate          CertTemplate,
    rsaKeyLen             INTEGER OPTIONAL,
}
```

< TBD: The OID id-it-caCerts, id-it-rootCaKeyUpdate, and id-it-certReqTemplate must be defined by IANA >

Appendix B. Example for CertReqTemplate

This Section provides a concrete example for the content of an infoValue used of type id-it-certReqTemplate as described in Section 4.4.4.

Suppose the server requires that the certTemplate contains the issuer field with a value to be filled in by the EE, the subject field with a common name to be filled in by the EE and two organizational unit fields with given values "myDept" and "myGroup", the publicKey field with an RSA public key of length 2048, the subjectAltName extension with DNS name "www.myServer.com" and an IP address to be filled in, the keyUsage extension marked critical with the value

digitalSignature and keyAgreement, and the extKeyUsage extension with values to be filled in by the EE. Then the infoValue with certTemplate and rsaKeyLen returned to the EE must be encoded as follows:

```
SEQUENCE {
  SEQUENCE {
    [3] {
      SEQUENCE {}
      }
    [5] {
      SEQUENCE {
        SET {
          SEQUENCE {
            OBJECT IDENTIFIER commonName (2 5 4 3)
            UTF8String ''
            }
          }
          SEQUENCE {
            OBJECT IDENTIFIER organizationalUnitName (2 5 4 11)
            UTF8String 'myDept'
            }
          }
        SET {
          SEQUENCE {
            OBJECT IDENTIFIER organizationalUnitName (2 5 4 11)
            UTF8String 'myGroup'
            }
          }
        }
      }
    [6] {
      SEQUENCE {
        OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
        NULL
        }
      BIT STRING, encapsulates {
        SEQUENCE {}
        }
      }
    [9] {
      SEQUENCE {
        OBJECT IDENTIFIER subjectAltName (2 5 29 17)
        OCTET STRING, encapsulates {
          SEQUENCE {
            [2] 'www.myServer.com'
            [7] ''
            }
```

```
            }
          }
        SEQUENCE {
          OBJECT IDENTIFIER keyUsage (2 5 29 15)
          BOOLEAN TRUE
          OCTET STRING, encapsulates {
            BIT STRING 3 unused bits
              '10001'B
            }
          }
        SEQUENCE {
          OBJECT IDENTIFIER extKeyUsage (2 5 29 37)
          OCTET STRING, encapsulates {
            SEQUENCE {}
            }
          }
        }
      }
    INTEGER 2048
    }
```

Appendix C. History of changes

Note: This section will be deleted in the final version of the document.

From version 01 -> 02:

- Extend Section 1.4 with regard to conflicts with UNISIG Subset-137.
- Minor clarifications on extraCerts in Section 3.3 and Section 4.1.1.
- Complete specification of requesting a certificate from a trusted PKI with signature protection in Section 4.1.2.
- Changed from symmetric key-encryption to password-based key management technique in section Section 4.1.6.3 as discussed on the mailing list (see thread "draft-ietf-lamps-lightweight-cmp-profile-01, section 5.1.6.1")
- Changed delayed enrollment described in Section 4.1.7 from recommended to optional as decided at IETF 107
- Introduced the new RootCAKeyUpdate structure for root CA certificate update in Section 4.4.3 as decided at IETF 107 (also

see email thread "draft-ietf-lamps-lightweight-cmp-profile-01, section 5.4.3")

- Extend the description of the CertReqTemplate PKI management operation, including an example added in the Appendix. Keep rsaKeyLen as a single integer value in Section 4.4.4 as discussed on the mailing list (see thread "draft-ietf-lamps-lightweight-cmp-profile-01, section 5.4.4")

- Deleted Sections "Get certificate management configuration" and "Get enrollment voucher" as decided at IETF 107

- Complete specification of adding an additional protection by an PKI management entity in Section 5.1.3.

- Added Section 6.1 and extended Section 6.2 on definition and discovery of supported HTTP URIs and content types, add a path for nested messages as specified in Section 5.1.3 and delete the paths for /getCertMgtConfig and /getVoucher

- Changed Section 6.5 to address offline transport and added more detailed specification file-based transport of CMP

- Added a reference to the new I-D of Mohit Sahni on "CoAP Transport for CMPV2" in Section 6.6; thanks to Mohit supporting the effort to ease utilization of CMP

- Moved the change history to the Appendix

- Minor changes in wording

From version 00 -> 01:

- Harmonize terminology with CMP [RFC4210], e.g.,

  * transaction, message sequence, exchange, use case -> PKI management operation

  * PKI component, (L)RA/CA -> PKI management entity

- Minor changes in wording

From draft-brockhaus-lamps-lightweight-cmp-profile-03 -> draft-ietf-lamps-lightweight-cmp-profile-00:

- Changes required to reflect WG adoption

- Minor changes in wording

From version 02 -> 03:

- Added a short summary of [RFC4210] Appendix D and E in Section 1.3.
- Clarified some references to different sections and added some clarification in response to feedback from Michael Richardson and Tomas Gustavsson.
- Added an additional label to the operational path to address multiple CAs or certificate profiles in Section 6.2.

From version 01 -> 02:

- Added some clarification on the key management techniques for protection of centrally generated keys in Section 4.1.6.
- Added some clarifications on the certificates for root CA certificate update in Section 4.4.3.
- Added a section to specify the usage of nested messages for RAs to add an additional protection for further discussion, see Section 5.1.3.
- Added a table containing endpoints for HTTP transport in Section 6.2 to simplify addressing PKI management entities.
- Added some ToDos resulting from discussion with Tomas Gustavsson.
- Minor clarifications and changes in wording.

From version 00 -> 01:

- Added a section to specify the enrollment with an already trusted PKI for further discussion, see Section 4.1.2.
- Complete specification of requesting a certificate from a legacy PKI using a PKCS#10 [RFC2986] request in Section 4.1.5.
- Complete specification of adding central generation of a key pair on behalf of an end entity in Section 4.1.6.
- Complete specification of handling delayed enrollment due to asynchronous message delivery in Section 4.1.7.
- Complete specification of additional support messages, e.g., to update a Root CA certificate or to request an RFC 8366 [RFC8366] voucher, in Section 4.4.

- Minor changes in wording.

From draft-brockhaus-lamps-industrial-cmp-profile-00 -> draft-brockhaus-lamps-lightweight-cmp-profile-00:

- Change focus from industrial to more multi-purpose use cases and lightweight CMP profile.
- Incorporate the omitted confirmation into the header specified in Section 3.1 and described in the standard enrollment use case in Section 4.1.1 due to discussion with Tomas Gustavsson.
- Change from OPTIONAL to RECOMMENDED for use case 'Revoke another's entities certificate' in Section 5.2, because it is regarded as important functionality in many environments to enable the management station to revoke EE certificates.
- Complete the specification of the revocation message flow in Section 4.2 and Section 5.2.
- The CoAP based transport mechanism and piggybacking of CMP messages on top of other reliable transport protocols is out of scope of this document and would need to be specified in another document.
- Further minor changes in wording.

Authors' Addresses

Hendrik Brockhaus
Siemens AG

Email: hendrik.brockhaus@siemens.com

Steffen Fries
Siemens AG

Email: steffen.fries@siemens.com

David von Oheimb
Siemens AG

Email: david.von.oheimb@siemens.com